



المبادئ الأساسية للبرمجة
بلغة

Delphi 6.0

مهندس

شريف فتحى الشافعى

المبادئ الأساسية للبرمجة
بلغة

Delphi 6.0



مهندس
شريف فتحى الشافعى

٢٠٠٣

رقم الإيداع بدار الكتب : ٢٠٠٢/١٧٥٢٨
التقديم الدولي : ٣-٢٧٧-٢٨٧-٩٧٧

© حقوق النشر والطبع والتوزيع محفوظة لدار الكتب العلمية للنشر والتوزيع - ٢٠٠٣

لا يجوز نشر جزء من هذا الكتاب أو إعادة طبعه أو اختصاره بقصد الطباعة أو
اختزان مادته العلمية أو نقله بأي طريقة سواء كانت إلكترونية أو ميكانيكية أو بالتصوير أو
خلاف ذلك دون موافقة خطيه من الناشر مقدماً .

دار الكتب العلمية للنشر والتوزيع

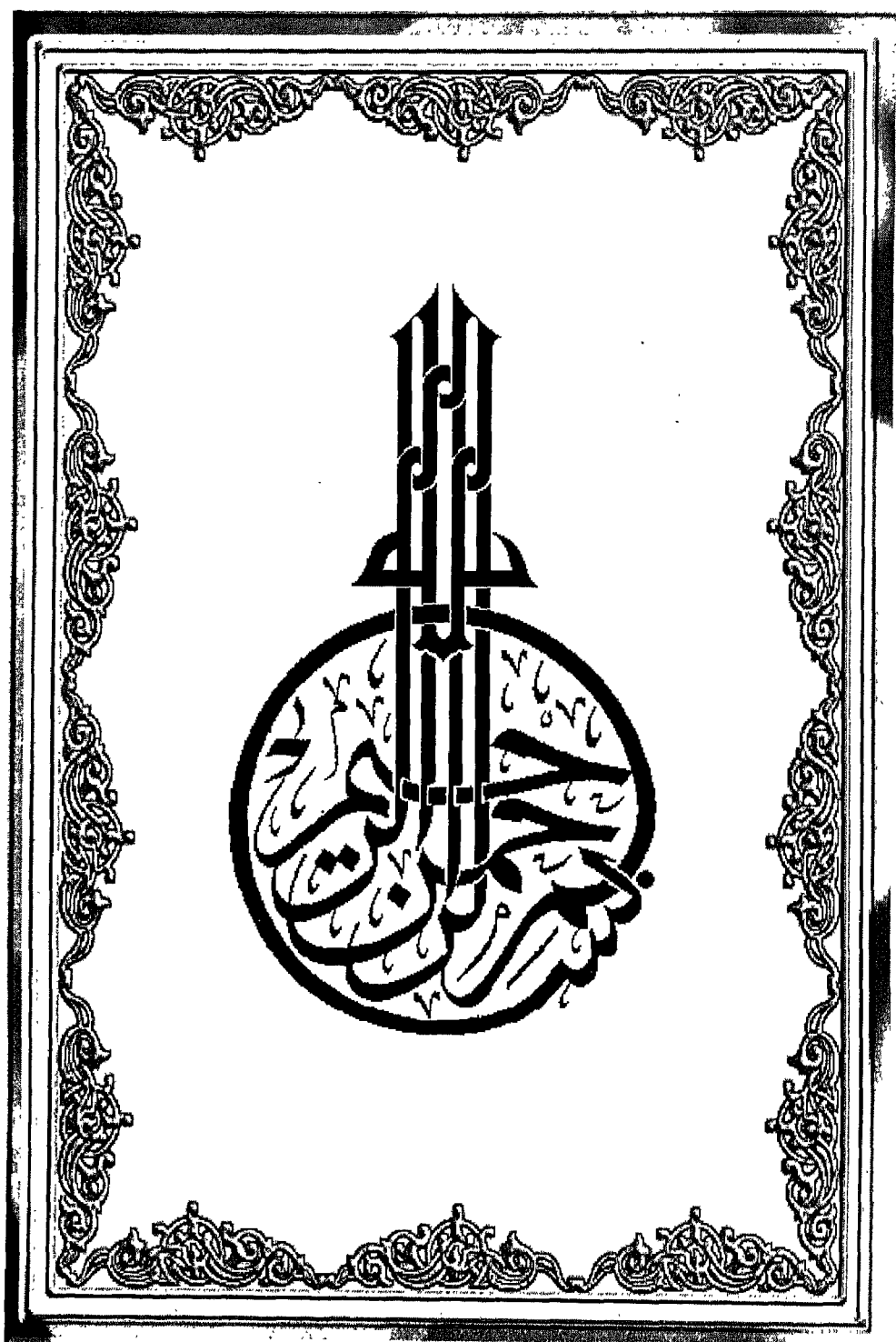
٥٠ شارع الشيخ ربحان - عابدين - القاهرة

٧٩٥٤٢٢٩ ☎

لتزيد من المعلومات يرجى زيارة موقعنا على الإنترنت

www.scientificbookhouse.com

e-mail: sbh@link.net



الإهداء

إلى زوجتى الحبيبة الغالية :

التي كان لها فضل كبير فى خروج هذا الكتاب إلى النور.

الفصل الأول

الجديد فى **Delphi 6**



ما هو الجديد في Delphi 6 ؟

من تعامل مع الإصدارات السابقة للغة البرمجة Delphi ثم أصبح الآن يتعامل مع الإصدار السادس لهذه اللغة فسيجد أن Delphi 6 تعمل على تقديم العديد من المظاهر والإمكانيات كما إنها تشتمل على كم هائل من التحسينات. وهذه الإمكانيات والتحسينات تتركز في المناطق التالية :

- بيئة العمل IDE (اختصار للمصطلح Integrated Development Environment) والذي يعنى بيئة التطوير المتكاملة).
- إعداد تطبيقات الإنترنت.
- التفاعل مع لغة البرمجة XML.
- المترجم Compiler الخاص باللغة.
- التعامل مع أدوات التحكم ActiveX والمكونات COM.
- الدعم المقدم لتطبيقات قواعد البيانات.
- إمكانية أو الخاصية CORBA.
- إعداد الأفعال Actions التى تستجيب من خلالها التطبيقات لطلبات المستخدمين.
- المتغيرات الخاصة أو المفصلة Custom.
- وحدات وإمكانيات الـ VCL.
- وحدات وإمكانيات الـ RTL.
- إعداد تطبيقات وبرامج لديها القدرة على العمل بمختلف نظم التشغيل.
- أدوات الترجمة.
- التغييرات التى أجريت على طريقة نشر وتوزيع التطبيقات والبرامج.
- حساسية نظام المساعدة.

بعض من الإمكانيات والتحسينات تكون غير متاحة بكافة إصدارات
Delphi editions لغة .





المظاهر والتحسينات الجديدة ببيئة التطوير المتكاملة IDE

لقد نالت بيئة التطوير المتكاملة IDE قدر كبير من المظاهر والإمكانيات والتحسينات الجديدة والتي تركزت فى الآتى :

- الـ Modules الخاصة بالبيانات.
- المشهد الشجرى أو الهرمى TreeView لخصائص وصفات الكائنات.
- محرر الكود البرمجى.
- أداة فحص الكائن.
- أدوات مشاهدة الكود البرمجى.
- مدير المشروع.
- قائمة الملف.
- صندوق حوار العناصر الجديدة New Items.
- شريط أدوات الإنترنت.
- التغييرات التى أجريت على بالليته المكونات.
- الـ modules الخاصة بإعداد خريطة للمفاتيح.
- صندوق حوار خيارات بيئة العمل Environment Options.
- صندوق حوار الفهارس Directories.
- طريقة عرض محتويات القوائم.
- كتابة حزم برمجية فى مرحلة تصميم التطبيقات Design-time packages.
- الـ Modules الخاصة بالبيانات (كافة الإصدارات)

أداة تصميم Module البيانات والتي كانت متوفرة لدى Delphi 5 تم تقسيمها إلى ثلاثة أجزاء وهى :

- (١) Module البيانات القياسى (كما هو الحال فى Delphi 4) وهو يعتبر بديل لصفحة المكونات Components page.



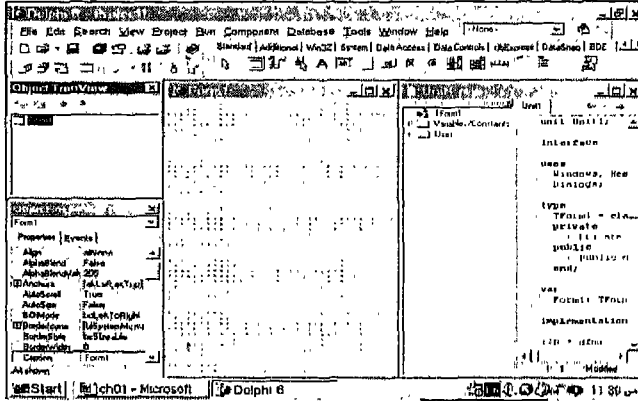
(٢) أداة المشاهدة الشجرية TreeView للكائن التي تعتبر بديل للقسم الأيسر بأداة تصميم Module البيانات.

(٣) صفحة الديجرام Diagram Page التي تعتبر بديل لصفحة ديجرام البيانات.

أداة المشاهدة الشجرية TreeView للكائن (كافة الإصدارات)

هذه الأداة تقع في الركن الأيسر العلوي لبيئة التطوير المتكاملة IDE كما هو موضح

بالشكل التالي :



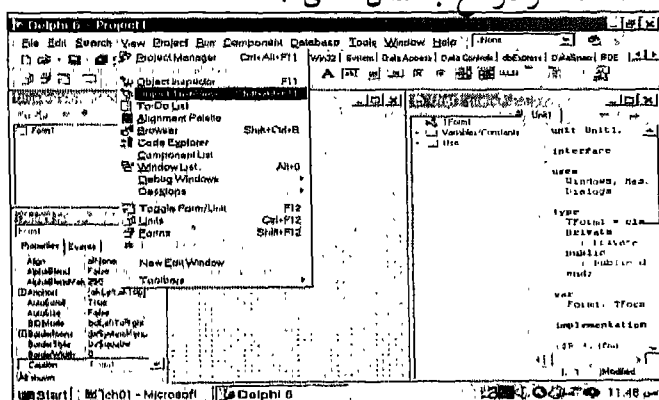
شكل توضيحي :

هذا وأداة المشاهدة الشجرية TreeView تعتبر ديجرام شجري يقوم بعرض العلاقات التبادلية المنطقية بين المكونات المرئية والغير مرئية الموجودة بفورمة أو بـ Module بيانات أو بإطار Frame.

تعمل أداة المشاهدة الشجرية TreeView بطريقة متزامنة ولحظية مع أداة فحص الكائن وأداة تصميم الفورمة ومن ثم لو أنك اخترت عنصر وقمت بتغيير المشاهدة إلى أى من الأدوات الثلاثة سالفة الذكر ستجد أن المشاهدة تتغير في نفس الوقت في الإداتين الأخرتين. وأكثر من ذلك نقول أن صفحة الديجرام الموجودة في محرر الكود البرمجي تعمل فقط عن طريق سحب وإسقاط العناصر من النافذة الخاصة بأداة المشاهدة الشجرية TreeView.

لقد تم إضافة المزيد من المهام الوظيفية لأداة المشاهدة الشجرية TreeView ومن بين هذه المهام الوظيفية ما يلي :


لقد تم وضع نافذة أداة المشاهدة الشجرية TreeView فوق نافذة أداة فحص الكائن Object Inspector وإذا لم تكن ظاهرة على الشاشة في مرحلة ما فإنها تظهر فوراً عندما تضغط على مجموعة المفاتيح Alt+Shift+F11 معاً أو عندما تختار Object TreeView من القائمة View كما هو موضح بالشكل التالي :



شكل توضيحي :

تتضمن أداة المشاهدة الشجرية TreeView المكونات الموجودة بكل من الفورمة والإطار Module والبيانات.

تتضمن أداة المشاهدة الشجرية TreeView المكونات المرئية والغير مرئية أيضاً.

نافذة أداة المشاهدة الشجرية TreeView تتضمن شريط أدوات جديد يشتمل على الأيقونة New Item والأيقونة Delete  والأيقونة Move Up والأيقونة Move Down. وهذه الأيقونات تعمل من أجل خصائص المكون. فعلى سبيل المثال لو أنك أضفت مكون Dataset فإنك تستطيع اختيار الخاصية Aggregate ثم تنقر بالماوس على الأيقونة New Item لكي تضيف حقل جديد. ولو أن هناك أشياء مختلفة الأنواع يمكن إضافتها في هذه الحالة سيتم فتح قائمة صغيرة عند النقر بالماوس على الأيقونة New Item.

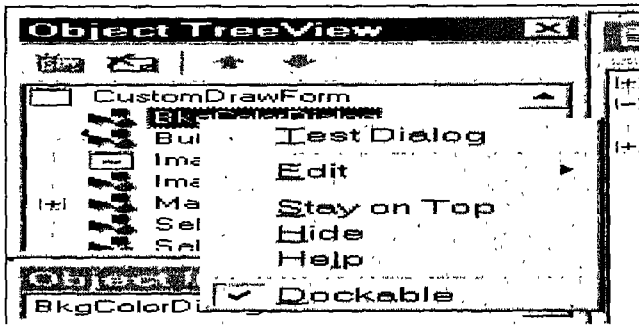
لقد تم إضافة المزيد من glyphs التي تعمل على عرض أو تمثيل نوع العنصر الذي قد يكون أب Parent (أصل) مرئى أو قد يكون مكون وليد أو تابع Child أو قد يكون مكون غير مرئى أو قد يكون خاصية ضمنية أو غير ضمنية. فعلى سبيل المثال العناصر التي يتم إنشاؤها ضمناً بدون أن تشعر أنت مثل التعامل الافتراضى



default session تظهر باللون الأبيض والأسود فى نافذة أداة المشاهدة الشجرية
TreeView.

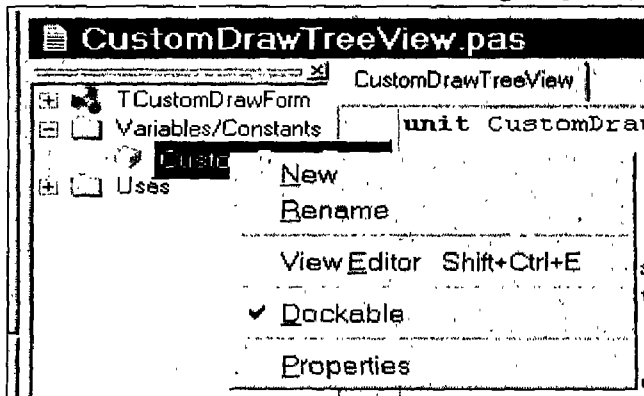
❶ فى الإصدارات السابقة كنا نجد أنه فى أثناء كون قيم خصائص أى مكون غير مكتملة نلاحظ أن glyph الخاص بهذا المكون تكون محاطة بدائرة حمراء. أما الآن فإننا نلاحظ علامة استفهام حمراء داخل دائرة صفراء تظهر على يسار الأيقونة.

❷ النقر بالزر الأيمن للماوس على أى عنصر من العناصر الموجودة بنافذة أداة المشاهدة الشجرية TreeView يؤدي إلى ظهور قائمة مختصرة تضم الخيارات الموضح فى الشكل التالى :



شكل توضيحي :

وهذه الخيارات نلاحظ أنها أكثر من التى تشتمل عليها القائمة المختصرة التى تظهر عند النقر بالزر الأيمن للماوس على أى عنصر بنافذة أداة تصميم Module البيانات كما هو موضح بالشكل التالى :



شكل توضيحي :



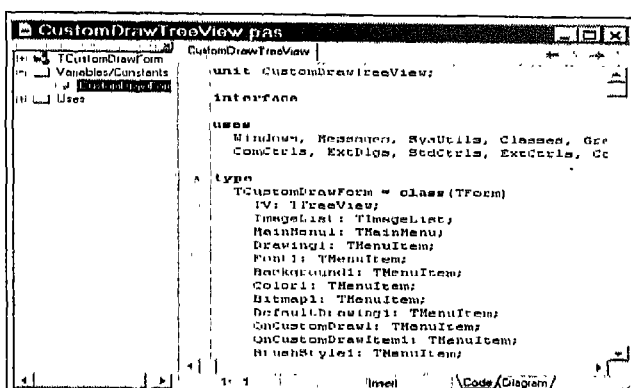
محرر الكود البرمجي

فيما يلي سوف نستعرض سويا المظاهر الجديدة التي أضيفت لمحرر الكود البرمجي من خلال Delphi 6.

أدوات تصميم السطح (الإصدار الفني Professional والإصدار المتكامل Enterprise)

محرر الكود البرمجي Code Editor أصبح الآن يقدم الدعم الكامل لأدوات تصميم السطح Surface designers أو للحزمة المحملة للمشاهد الخاصة أو المفصلة. وهذه المشاهد يتم الوصول إليها من خلال مجموعة من التبويبات الموجودة في سطر الحالة كما هو موضح بالشكل التالي :

شكل توضيحي :



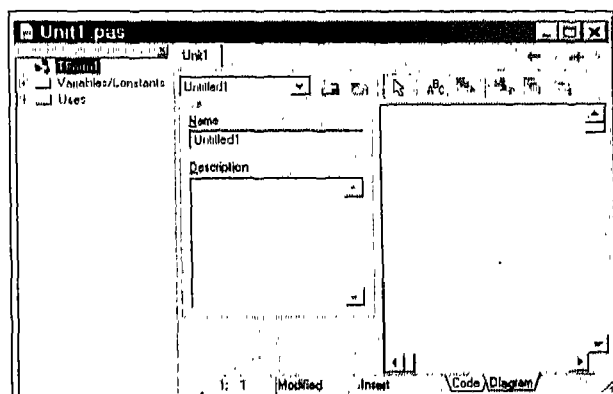
المشهد الأساسي built-in الوحيد هو صفحة الكود البرمجي القياسية. ونود هنا القول بأنه بناء على الإصدار الذي تتعامل معه من لغة Delphi فإنك تستطيع الوصول للصفحات التالية بمحرر الكود البرمجي :

صفحة الديجرام (الإصدار الفني Professional والإصدار المتكامل Enterprise)

الشكل التالي يوضح لنا صفحة الديجرام الموجودة في محرر الكود البرمجي Code

Editor

شكل توضيحي :



وهذه الصفحة تعمل على توفير العديد من الأدوات المرئية Visual من أجل إعداد وتهيئة ديجرام من المستطيلات والمربعات والخطوط لعرض العلاقات التبادلية بين المكونات المرئية والمكونات غير المرئية. ويمكن اعتبار صفحة الديجرام بمثابة أداة توثيق حيث إنها تقوم بتوضيح هذه العلاقات التبادلية تخطيطيا Schematically كما إنها تسمح لك بأن تضيف أى تعليقات ترغبها للديجرام. ونود هنا القول بأن المكونات لا تظهر فى صفحة الديجرام حتى تقوم برسمهم من خلال أداة المشاهدة الشجرية TreeView للكائن.

المظاهر والإمكانات الجديدة لدى صفحة الديجرام :

- تستطيع أن تختار العديد من العناصر فى نفس الوقت من نافذة المشاهدة الشجرية للكائن Object TreeView ثم تسحب هذه العناصر إلى داخل صفحة الديجرام.
- الجانب الأيسر بصفحة الديجرام أصبح مشتملا على حقل تستطيع فيه أن تكتب اسم ووصف لكل ديجرام تقوم بإنشاؤه كما إنه يشتمل أيضا على قائمة منسدلة للعثور على الديجرامات التى تم إنشاؤها وتسميتها من قبل.
- لقد تم إضافة شريط الأدوات يضم العديد من الأيقونات إلى صفحة الديجرام. وهذه الأيقونات لعمل الوصلات بين العلاقات التبادلية ولتحديد مواضع بلوكات تعليق داخل الديجرام.
- يتم تلقائيا تخصيص عناوين لأدوات وصل الصفات Property connectors. وأنت تستطيع سحب العناوين وتسقطها فى أى موضع بصفحة الديجرام.
- تستطيع أن تنشأ ديجرام لكل Module للبيانات أو لكل فورمة أو لكل إطار قمت بإضافته إلى المشروع.

صفحات الـ WebSnap (الإصدار المتكامل Enterprise)

عند إنشاء تطبيق من أجل أن يعمل فى أحد خوادم الويب باستخدام الـ WebSnap فإن المكونات التى توجد فى module صفحة الويب تعمل على تكوين صفحات لكل من الآتى :

- الـ Script المعد بلغة HTML.
- ناتج تنفيذ الكود البرمجى المكتوب بلغة HTML.

المعاينة Preview.

التسلسل الشجري XML Tree.

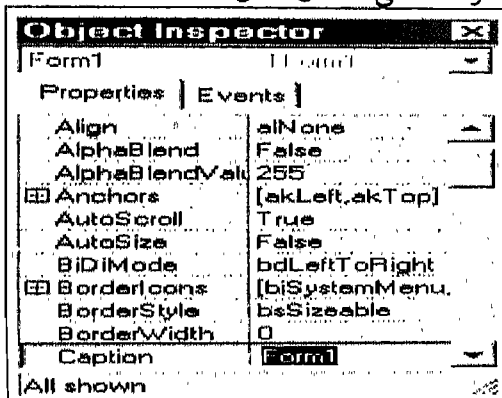
التسلسل الشجري XSL Tree.

تبويبات السحب والإسقاط (كافة الإصدارات)

تبويبات السحب والإسقاط عبارة عن وحدة التبويبات Tabs الموجودة في الجانب العلوي بمحرر الكود البرمجي والتي يمكن الشعور بها من خلال سحب وإسقاط هذه التبويبات. فعلى سبيل المثال لو أن لديك وحدتين إحداهما تسمى About والأخرى تسمى TextEditor في هذه الحالة تستطيع سحب الوحدة About إلى الجانب الأيمن للوحدة TextEditor.

أداة فحص الكائن Object Inspector (كافة الإصدارات)

نافذة أداة فحص الكائن Object Inspector أصبحت الآن موجودة أسفل نافذة المشاهدة الشجرية للكائن Object Tree وهي موضحة في الشكل التالي :



شكل توضيحي :

فيما يلي سوف نستعرض سويًا المظاهر والإمكانيات الجديدة التي أضيفت لأداة فحص الكائن Object Inspector من خلال Delphi 6.

قائمة عرض اللحظة Instance list box

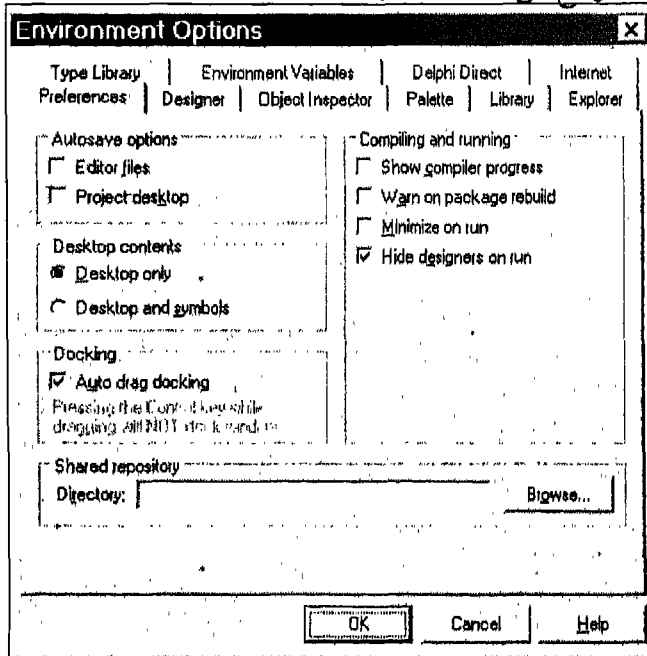
لقد تم إضافة قائمة منسدلة لحظة بالجزء العلوي لنافذة أداة فحص الكائن Object Inspector وهي تتمتع بالمظاهر والإمكانيات التالية :



- ❶ تعمل القائمة اللحظية على استعراض اسم القطاع لكل كائن موجودة في القائمة وليس فقط للكائن الموجود في قمة القائمة.
- ❷ تستطيع أن تمنح أى مكون نفس الأسم الممنوح لأى فورمة أو Module بيانات موجود به. فعلى سبيل المثال تستطيع أن تضيف مكون أيقونة إلى الفورمة Form1 ثم تقوم بتغيير اسمه ليصبح Form1 أيضا وفي هذه الحالة كلا الأسمين سيظهران في القائمة اللحظية.
- ❸ تقوم القائمة اللحظية بعرض أداة تعليق Tooltip للمكون المختار حاليا ويعتبر ذلك مفيدا فى حالة أن اسم المكون كان أعرض من القائمة اللحظية.
- ❹ يمكن إخفاء القائمة اللحظية.

صندوق حوار الخصائص

قائمة المحتوى Context الخاصة بنافذة أداة فحص الكائن Object Inspector أصبح لديها صندوق حوار خصائص جديد والذي يمكن الوصول إليه أيضا عن طريق فتح القائمة Tools واختيار منها الأمر Environment Options ليظهر على الشاشة صندوق الحوار Environment Options الموضح فى الشكل التالى :



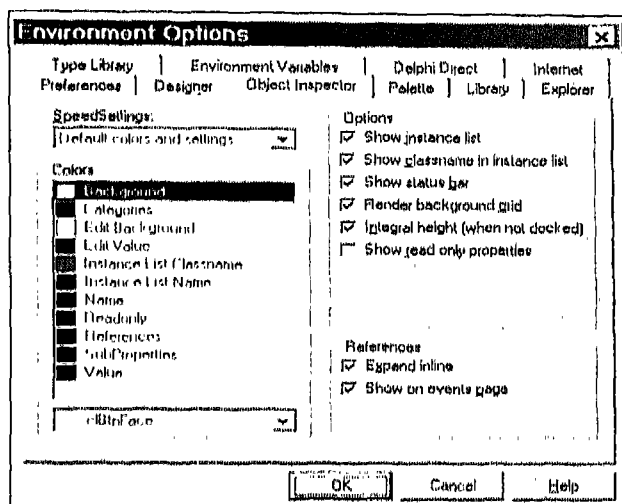
شكل توضيحي :



وفيه انقر بالماوس على التبويب Object Inspector ليظهر على السطح كما هو

موضح بالشكل التالي :

شكل توضيحي :



وهذا التبويب يضم العديد من خيارات العرض التالية :

مجموعة الـ SpeedSettings التي تعمل على تفصيل الألوان الخاصة بأداة فحص الكائن Object Inspector.

عرض أو إخفاء كل من القائمة اللحظية وأسماء القطاعات في القائمة اللحظية وشريط الحالة وشبكة نقاط الخلفية والخصائص التي تكون للقراءة فقط.

خصائص مرجعيات الكائن يمكن تمديدتها inline وعرضها في كل من صفحة الخصائص وصفحة الأحداث Events.

مرجعيات المكون الممتدة Inline

تعمل مرجعيات المكون الممتدة Inline على عرض كل من الخصائص والأحداث الخاصة بأي كان مرجعي وذلك بدون الحاجة للقيام فعليا باختيار المكون المرجعي نفسه. وفي هذا الجزء نجد المظاهر والإمكانات الجديدة التالية :

الخصائص التي تشير مرجعيا لكائن درجة ثانية تكون حمراء اللون كما أن الخصائص الخاصة بهذا الكائن الدرجة الثانية تكون بطبيعة الحال خضراء اللون.



الأحداث التى تشير مرجعيا لكائن درجة ثانية تكون حمراء اللون كما أن الأحداث الخاصة بهذا الكائن الدرجة الثانية تكون بطبيعة الحال خضراء اللون.

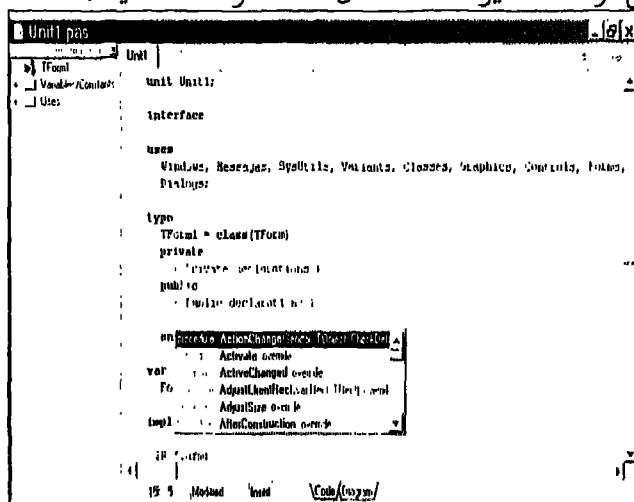
الخصائص التى تتفاعل مع بعضها البعض يمكن الإشارة المرجعية إليها inline.

أدوات التعامل مع الكود البرمجى Code insight tools (كافة الإصدارات)

إمكانية التعامل مع الكود البرمجى أصبحت لديها إمكانيات ومظاهر تم تحديثها وخاصة بالنسبة لتكملة الكود ومعاملات الكود code parameters.

تكملة الكود البرمجى Code completion

الآن عندما تقوم بنفسك باستدعاء خاصية تكملة الكود البرمجى (بالضغط على المفاتيح Ctrl+Space) بلوحة المفاتيح وأنت لاتزال فى سطر فارغ داخل جسم أى روتين فستجد أنه تظهر على الفور قائمة منسدلة (الموضح فى الشكل التالى) تقدم كل رموز من وحدات RTL الإضافية حتى لو كانت غير مستخدمه من خلال الوحدة الحالية :



شكل توضيحي :

فيما يلى سنستعرض سويا المظاهر والإمكانيات الجديدة بهذه الخاصية :

تصفية وفرز كافة الإعلانات عن طرق التفاعل interface التى يتم الإشارة إليها مرجعيا من خلال قطاعات صفة القراءة أو الكتابة. فالقائمة المنسدلة تعرض فقط الخصائص والطرق المستقلة stand-alone التى تم الإعلان عنها فى نوعية التفاعل interface type.

- إجراء عملية تصفية وفرز فى أثناء الكتابة بمحرر الكود البرمجى. كما أن النص يتم إزالته القائيا عندما تختار أى عنصر من القائمة المنسدلة.
- العمل داخل منطقة تعريف القطاع Class فى قسم التفاعل مع تقديم الدعم لإمكانية الاختيار الكتعدد multiselect.
- عرض الطرق المتوارثة والتصورية بالإضافة إلى طرق التفاعل والخصائص.
- إمكانية تغيير حجم القائمة المنسدلة.
- لقد تم إضافة الألوان للمساعدة فى التمييز بين مختلف العناصر الموجودة بالقائمة المنسدلة. فعلى سبيل المثال الإجراءات تكون ملونة باللون الأزرق المخضر teal فى حين أن الدوال تكون ملونة بالأزرق الغامق. ولكن على العموم تستطيع أن تقوم بتغيير الألوان الافتراضية فى أحقية الاستخدام Registry.
- تقديم الدعم لأساليب الرسائل WM_XXX و CM_XXX و CN_XXX المعتمدة على الثوابت المتشابهة الأسماء والموجودة بكافة الوحدات بالجملة Uses.
- أساليب الاستخلاص أصبحت الآن تظهر باللون الأحمر فى مناطق الإعلانات عن الأنواع Type declarations. أما باقى كافة الـ contexts الأخرى الخاصة بأدوات التعامل مع الكود البرمجى تعمل على إظهار أساليب الاستخلاص باستخدام الألوان المعتادة لكل من الإجراءات والدوال.

معاملات الكود البرمجى code parameters

- لو أن عنصر كان عبارة عن إجراء أو دالة يستخدم عدد من المعاملات parameters فى هذه الحالة يتم إضافة "" إلى اسم هذا العنصر كما يتم عرض على الفور أداة تعليق مشتملة على الكود البرمجى للمعامل.

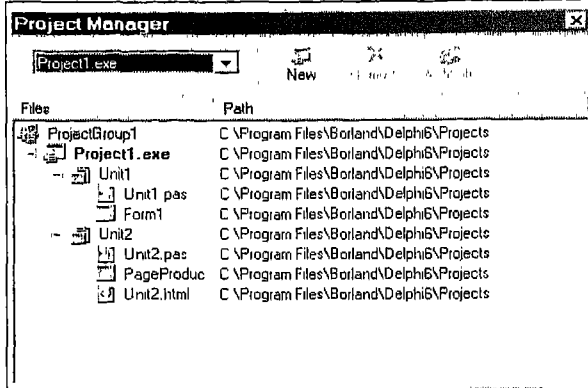
المودول الجديد لتخطيط المفاتيح Key mapping (كافة الإصدارات)

- لقد تم إضافة مجموعة تخطيط المفاتيح الخاصة بلغة VB (Visual Basic) إلى الـ modules الخاصة بتخطيط المفاتيح الموجودة بالفعل لدى لغة Delphi.



الحزم Packages في مدير المشروع (كافة الإصدارات)

الآن نجد أن كافة حزم المشاريع المفتوحة حاليا تكون معروضة في مشهد مدير المشروع كما هو موضح بالشكل التالي :

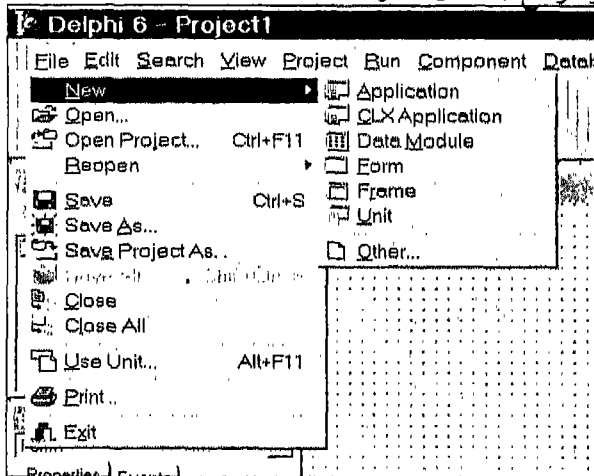


شكل توضيحي :

ونود هنا القول بأن هذه nodes التي تشير مرجعيا إلى حزم المشروعات يمكن استخدامها للمساعدة في متابعة وإدارة المشروع النشط حاليا كما يمكن الاستعانة بها أيضا كأداة للتجول لكي تصل إلى أى حزمة مفتوحة حتى عندما تكون هذه الحزمة ليست عضوا في مجموعة المشروع الحالي.

قائمة الملف File Menu

لقد تم تحديث قائمة الملف File Menu وهذا التحديث يتمثل في إضافة القائمة الفرعية New للقائمة File كما هو موضح بالشكل التالي :



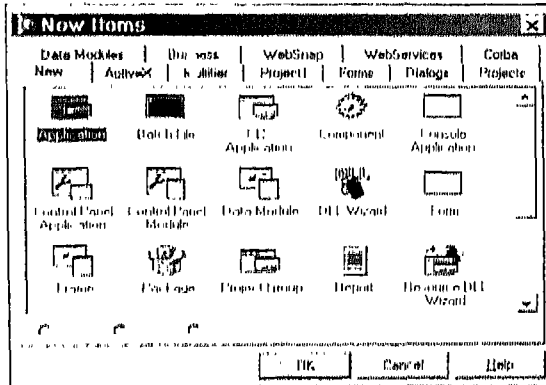
شكل توضيحي :



وكما نرى فإن هذه القائمة الفرعية تشتمل على ٦ أنواع من المشاريع (تطبيق Application وتطبيق CLX وModule وبيانات وفورمة وإطار ووحدة) بالإضافة إلى باقي الاختيار Other الذى يعمل على فتح صندوق حوار العناصر الجديدة New Items.

صندوق حوار العناصر الجديدة New Items

الشكل التالى يوضح لنا صندوق حوار العناصر الجديدة New Items أو الذى يعرف بمخزن أو مستودع الكائنات Object Repository :

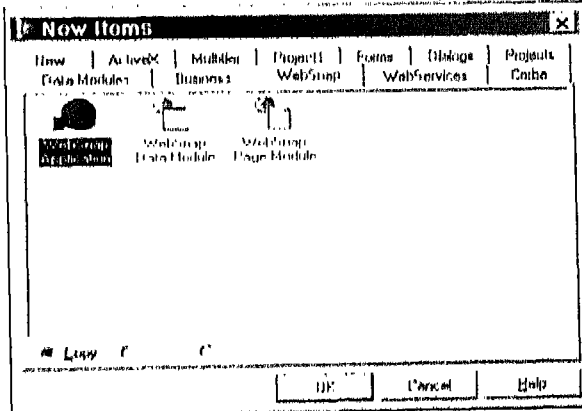


شكل توضيحي :

كما نرى أن صندوق الحوار New Items أصبح مشتملا على ثلاثة تبويبات جديدة بالإضافة إلى عدد من المعالجات Wizards الجديدة وعدد من modules البيانات الجديدة التالية :

التبويب WebSnap :

الشكل التالى يوضح لنا محتويات هذا التبويب الجديد :

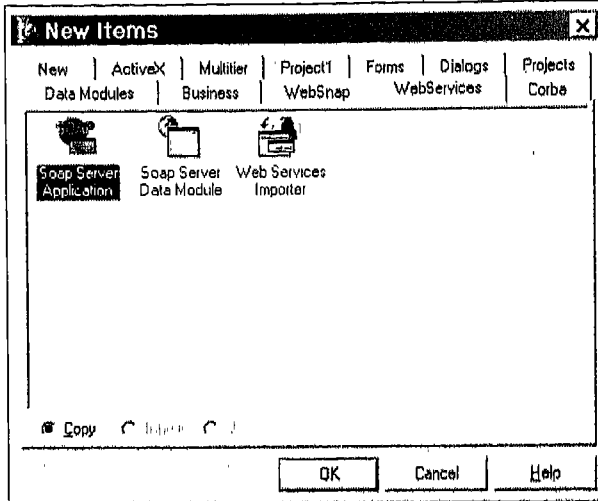


شكل توضيحي :



التبويب Web Services :

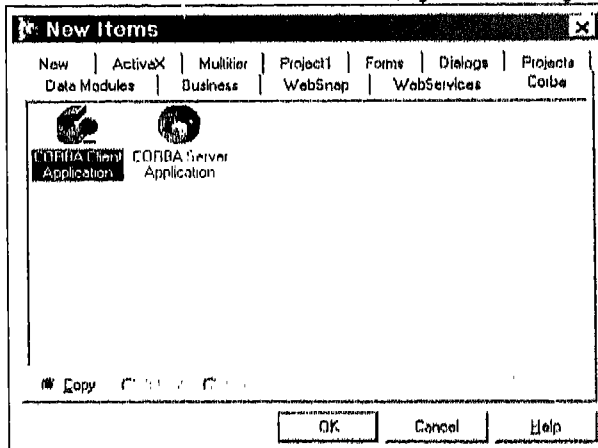
الشكل التالي يوضح لنا محتويات هذا التبويب الجديد :



شكل توضيحي :

التبويب CORBA :

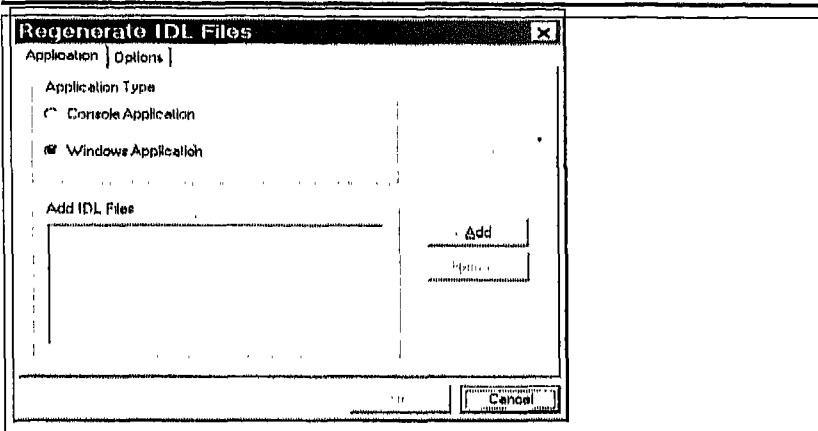
الشكل التالي يوضح لنا محتويات هذا التبويب الجديد :



شكل توضيحي :

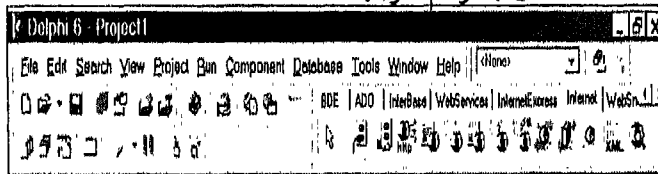
تستطيع أيضا أن تصل لهذه المعالجات من خلال الأمر regenerate CORBA IDL Files الموجود بالقائمة Tools والذي يؤدي لفتح صندوق الحوار Regenerate CORBA IDL Files الموضح في الشكل التالي :





شريط أدوات الإنترنت (الإصدار المتكامل Enterprise)

شريط أدوات الإنترنت (الموضح في الشكل التالي) يعتبر شريط أدوات جديد وهو يساعدك في إنشاء تطبيق WebSnap للعمل بخوادم الويب :



شكل توضيحي :

إذا كان شريط أدوات الإنترنت غير ظاهر على الشاشة في هذه الحالة افتح القائمة View ومنها اختر Toolbars ثم علم بالماوس على الاختيار Internet.



الأيقونات الموجودة بشريط أدوات الإنترنت تجعلك تصل إلى ثلاثة معالجات

جديدة وهي :

- المعالج New WebSnap App.
- المعالج New WebSnap Page Module.
- المعالج New WebSnap Data Module.
- المحرر الخارجي External Editor.

هذه الأيقونات تناظر الدوال المستخدمة دوما لإنشاء تطبيقات خوادم الويب من خلال WebSnap.





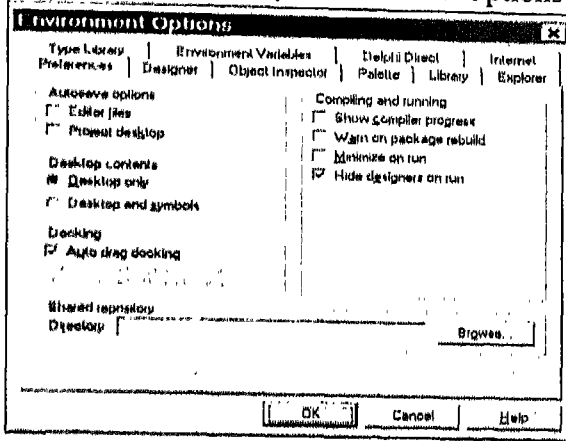
التغييرات التي أجريت على باليئة المكونات

- ❶ الصفحة الإضافية Additional Page أصبحت تمتلك الآن مكونات action band جديدة وذلك لتكوين قوائم الأفعال ومعالجتها وكذلك لبناء قوائم وشرائط أدوات مفصلة أو خاصة بما فيها كل من TActionManager و TCustomizedDlg و TActionToolBar و TActionMainMenuBar.
- ❷ بالنسبة للتطبيقات CLX يتم استخدام صفحة أدوات التحكم الشائعة الاستخدام Common Controls Page بدلا من الصفحة Win32 (الإصدار الفني Professional والإصدار المتكامل Enterprise).
- ❸ تم إضافة الصفحة WebSnap الجديدة والتي تشتمل على مكونات يمكن استخدامها لبناء تطبيقات خوادم الويب مع العديد من الـ Modules (الإصدار المتكامل Enterprise فقط).
- ❹ تم إضافة صفحة خدمات الويب WebServices Page لكي يتمكن المبرمج من كتابة تطبيقات متعددة الطبقات Multitier للعمل من خلال شبكة من الحاسبات الآلية أو من خلال شبكة الويب (الإصدار المتكامل Enterprise فقط).
- ❺ العديد من التغييرات تم إجراؤها على صفحات باليئة مكون قاعدة البيانات database Component palette pages (الإصدار الفني Professional والإصدار المتكامل Enterprise).
- ❻ تم إضافة الصفحة COM + Page الجديدة للمكون المسمى COMAdminCatalog (الإصدار الفني Professional والإصدار المتكامل Enterprise).
- ❼ تم إضافة ثلاث صفحات جديدة لبروتوكولات الاتصال عبر الإنترنت وهي الصفحة Indy Clients والصفحة Indy Servers والصفحة Indy Misc. ونود هنا القول بأن مكونات الإنترنت المباشرة (Indy) ما هي إلا بروتوكولات لفتح المصادر الإنترنت المتعددة والقائمة على أساس التكنولوجيا Blocking Sockets (الإصدار الفني Professional والإصدار المتكامل Enterprise).



صندوق حوار خيارات بيئة العمل Environment Options

الشكل التالي يوضح صندوق حوار خيارات بيئة العمل Environment Options (الذي يتم الوصول إليه من خلال الأمر Environment Options بالقائمة Tools):

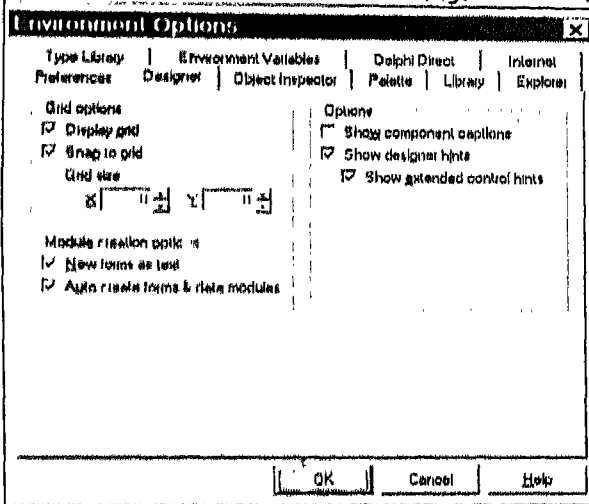


شكل توضيحي :

صندوق الحوار هذا أصبح الآن مشتملا على مجموعة التبويبات الجديدة التالية :

تبويب المصمم Designer :

الشكل التالي يقدم لنا محتويات هذا التبويب :



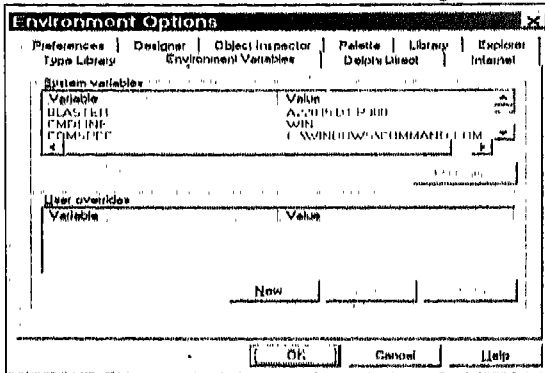
شكل توضيحي :

نلاحظ أنه قد تم نقل الخيارات الخاصة بمصمم الفورمة من التبويب Preferences إلى هذا التبويب كما نلاحظ أيضا وجود اختيار جديد وهو الاختيار Show extended control hints.



تبويب متغيرات بيئة العمل Environment Variables :

الشكل التالى يقدم لنا محتويات هذا التبويب :

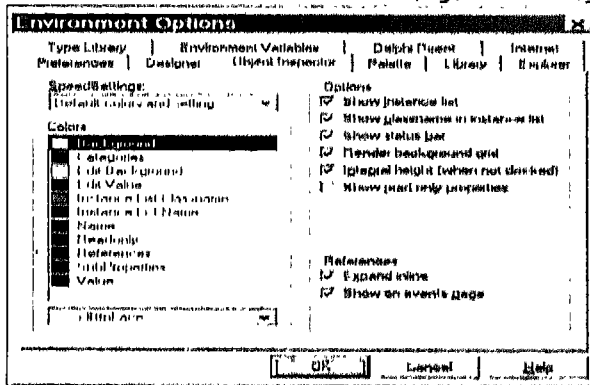


شكل توضيحي :

فى هذا التبويب تستطيع أن تشاهد المتغيرات الخاصة ببيئة العمل الخاصة باللغة كما تستطيع أيضا إنشاء وتحرير ومسح التعديلات التى قد يقوم بها المستخدم على متغيرات النظام.

تبويب أداة فحص الكائن Object Inspector :

الشكل التالى يقدم لنا محتويات هذا التبويب :



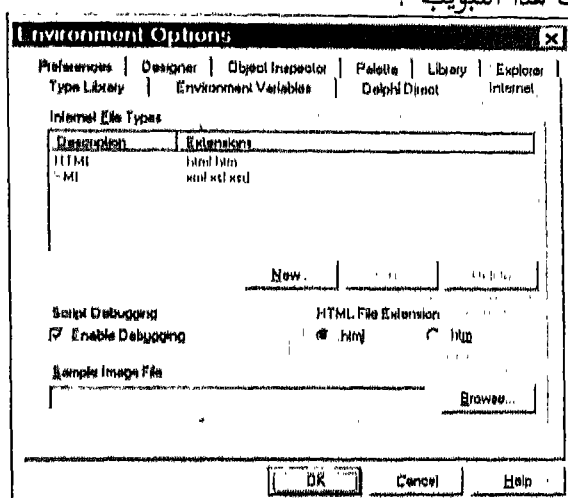
شكل توضيحي :

من خلال هذا التبويب تستطيع تحديد القيم المفضلة Preferences الخاصة بأداة فحص الكائن وذلك باستخدام صندوق الحوار Properties الجديد الذى يمكن الوصول إليه أيضا عن طريق النقر بالزر الأيمن للماوس على نافذة أداة فحص الكائن ثم اختيار Properties من القائمة المختصرة التى تظهر على الشاشة.



تبويب الإنترنت Internet :

الشكل التالي يقدم لنا محتويات هذا التبويب :



شكل توضيحي :

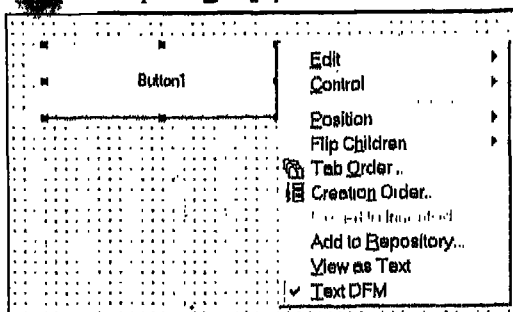
تستطيع من خلال هذا التبويب تحديد الأفضليات Preferences الخاصة بأنواع الملفات وخيارات الـ Script التي ترغب في استخدامها من أجل تطبيقات الـ WebSnap التي تتولى إعدادها.

صندوق حوار الفهارس Directories

صندوق حوار الفهارس Directories الخاص بتحرير العناصر أصبح الآن مثله مثل مكتبة المسار Library Path يقوم بعرض المسارات الغير صحيحة بلون رمادي (تكون غير نشطة). وأنت تستطيع إزالة المسارات الغير صحيحة من خلال المفتاح Delete Invalid Paths.

عرض قائمة المحتوى Context menu

في أثناء كون كائن مختار داخل فورمة أو في حالة كون نافذة مختارة أو في حالة كون دفة التحكم متركزة على أى عنصر في بيئة التطوير المتكاملة IDE، تستطيع الآن أن تعرض قوائم المحتوى الخاصة بالكائن أو العنصر المختار وذلك عن طريق الضغط على المفاتيح Alt+F10 بلوحة المفاتيح أو بالضغط على المفتاح Menu بلوحات مفاتيح مايكروسوفت. هذا وفيما يلي الشكل التالي يوضح لنا واحدة من قوائم المحتوى التي تظهر على الشاشة :



شكل توضيحي :

كتابة الحزم البرمجية في أثناء مرحلة التصميم Design-time packages

ينبغي عليك إضافة الملف DesignIDE.dcp إلى قائمة المتطلبات الخاصة بالحزمة البرمجية التي تود إعدادها.

مظاهر وإمكانات الإنترنت الجديدة

(الإصدار الفني Professional والإصدار المتكامل Enterprise)

فيما يلي سنتعرف سويا على مظاهر وإمكانات الإنترنت الجديدة المضافة حديثا للغة Delphi 6.

الدعم الخاص بخدمات الويب (الإصدار المتكامل Enterprise فقط)

الآن يمكن القول بأن المكونات والتغييرات التي أجريت على المترجم الخاص بلغة Delphi 6 تجعلنا قادرين على كتابة التطبيقات التي تعرف بأنها خدمات الويب كما تسمح لك بأن تقدم الدعم لكتابة تطبيقات clients التي تصل إلى خدمات الويب باستخدام SOAP.

خدمات الويب تعتبر تكنولوجيا جديدة ناشئة. وبسبب هذا فإن المكونات التي تعمل على تدعيم هذه الإمكانية أو التكنولوجيا بما فيها الوسائط Interfaces تكون معرضة للتغيير في أي وقت وباستمرار.



الدعم الخاص بتطبيقات خادم الويب

(الإصدار الفني Professional والإصدار المتكامل Enterprise)

يتضمن Delphi 6 مضيف لمظاهر الإنترنت الجديدة من أجل بناء تطبيقات خادم الويب بما فيها التغييرات التي تجرى على المظاهر والإمكانات WebBroker الموجودة بالفعل وأي مجموعة جديدة من المظاهر والإمكانات التي يطلق عليها WebSnap.



خادم الويب Apache

(الإصدار الفنى Professional والإصدار المتكامل Enterprise)

كل من الـ WebSnap والـ WebBroker أصبحا الآن يقدمان الدعم لخادم الويب Apache بالإضافة إلى تقديم الدعم لكل من ISAPI و CGI و WinCGI.

أداة معالجة الثغرات لتطبيقات الويب Web Application Debugger (الإصدار الفنى والإصدار المتكامل)

أداة معالجة الثغرات لتطبيقات الويب Web Application Debugger تمكنك من مراقبة الطلبات HTTP والاستجابات بالإضافة إلى الفترة الزمنية للاستجابة كما إنها تمكنك أيضا من تطوير وإعداد تطبيقات من خلال كل من الـ WebSnap والـ WebBroker وذلك بدون إقامة خادم ويب تجارى. ونود هنا القول بأن عملية معالجة الثغرات Debugging بأى تطبيق من تطبيقات خادم الويب يمكن أن تكون صعبة إلى حد ما وذلك بسبب أن التطبيق الذى تتولى إعداده يحتاج لأن يعمل من خلال استجابته لرسالة صادرة من أحد خوادم الويب. ومن ثم فإن أداة معالجة الثغرات لتطبيقات الويب تعمل على محاكاة مثل هذه الرسالة.

المظاهر والإمكانيات WebBroker المحسنة

(الإصدار الفنى Professional والإصدار المتكامل Enterprise)

الـ WebBroker أصبحت الآن :

- تعمل على تمكينك من كتابة تطبيقات خوادم الويب من أجل التطبيقات C.I.X.
- لديها القدرة على التوافق مع Modules الويب بكل من الإصدار الثالث والرابع والخامس من لغة Delphi مع وجود بعض القيود. فعلى سبيل المثال المشاريع القديمة قد تحتاج لإضافة مستخدمين جدد أو لتغيير مستخدمين حاليين. ونود هنا القول بأن Modules الويب المتعددة لن يتم تدعيمها بالتطبيقات المعدة بالإصدارات السابقة للغة Delphi.

مظاهر الـ WebSnap الجديدة (الإصدار المتكامل Enterprise فقط)

الـ WebSnap تكون معتمدة على مظاهر وإمكانيات الـ WebBroker لدى الإصدار الخامس للغة Delphi بالإضافة إلى الكثير من المظاهر والإمكانيات الوظيفية الجديدة والتى تتضمن الآتى :



● Modules المتعددة.

● المعالجات الجديدة.

● إعداد Scripts للعمل في خوادم الويب.

● المكونات الجديدة.

● أدوات تصميم السطح.

Modules الويب المتعددة

تطبيق واحد من تطبيقات المعدة من خلال WebSnap تكون لدية القدرة على تدعيم العديد من Modules الويب والتي تعمل على تقسيم التطبيق إلى وحدات كما إنها تسمح أيضا للعديد من المبرمجين بأن يعملوا بنفس المشروع مع التقليل بقدر الإمكان من التداخلات الغير مقصودة Conflicts. وأكثر من ذلك ما يلي:

● إمكانية تدعيم الحالات المتعددة Multiple Instance للطلبات المتلاحقة في فترة زمنية صغيرة إلى حد ما.

● تدعيم الإشارات المرجعية للكائنات بين Modules.

● يتم إنشاء Modules تلقائيا وذلك بهدف خدمة طلب أو عند الإشارة المرجعية لهم من خلال Module آخر.

المعالجات الجديدة لـ Module الويب

من خلال معالج تطبيق خادم الويب تستطيع أن تقوم ببناء تطبيق مع Module ويب مفصل أو مع Module تطبيق ويب أو مع Module صفحة الويب وذلك باستخدام المكونات الأربعة التالية :

● المكون TWebAppPageModule.

● المكون TwebAppDataModule.

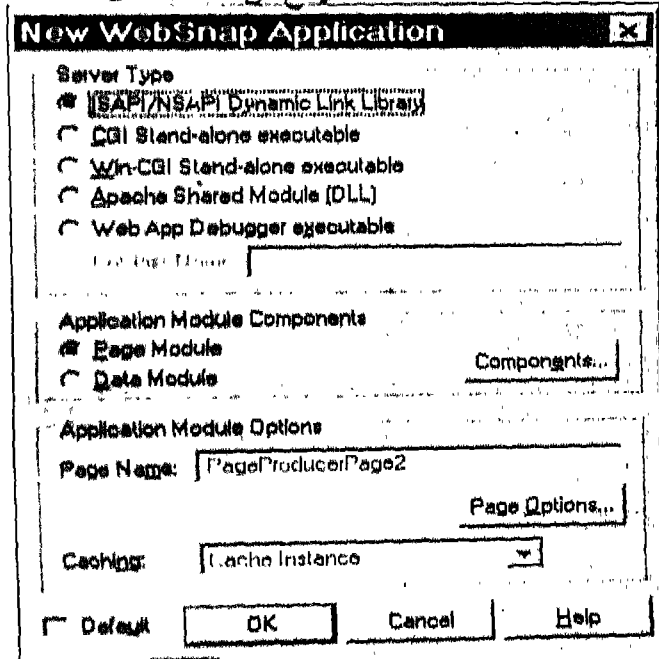
● المكون TWebPageModule.

● المكون TWebDataModule.

بالنسبة لـ Module بيانات الويب فهو يقوم بإنشاء حاوية Container للمكونات التي تكون متاحة للاستخدام المشترك عبر تطبيق الويب الذي تتولى إعداده. ولكن هذا الـ Module ليس لدية ملف قالب HTML، ملحق به.

أما Module صفحة الويب فيعمل على تعريف صفحة ويب جديدة بالتطبيق الذى تتولى إعداده. وهذا الـ Module لديه ملف قالب HTML ملحق به كما يعرض الـ Script الذى تتولى كتابته باستخدام أدوات تصميم سطح الـ WebSnap. كذلك فإن هذا الـ Module يعمل أيضا على تعريف اسم أى صفحة من الصفحات المنطقية وهو لا يحتاج لتعريف أى فعل action للوصول لصفحة الويب كما إنه يعمل أيضا على إدارة ملف القالب HTML من خلال نظام تصورى للملفات وعرضه بنافذة مدير المشروع.

هناك أكثر من طريقة لكى تصل لمجموعة معالجات Module الويب ومنها فتح القائمة File ثم اختيار New ثم Other وفى صندوق الحوار New Items انقر بالماوس على التبويب WebSnap ثم النقر المزدوج بالماوس على الأيقونة WebSnap Application ليظهر على الشاشة صندوق الحوار New WebSnap Application فى الشكل التالى :



شكل توضيحي :

هناك طريقة أخرى للقيام بنفس المهمة وهى فتح القائمة View ثم اختيار Toolbars ثم التعليم بالماوس على الاختيار Internet ثم النقر بالماوس على الأيقونة New WebSnap Application.



إعداد Scripts للعمل بخوادم الويب

تتضمن الـ WebSnap أداة إعداد Scripts للعمل بخوادم الويب وهى عبارة عن لغة بسيطة لإعداد الـ Scripts تم تصميمها خصيصا من أجل برمجة صفحات الويب. وهى تتميز بالآتى :

❖ إمكانية إعداد Scripts تكون لديها القدرة على الوصول للمكونات الخاصة بلغة Delphi.

❖ تقديم الدعم اللازم لمولد الـ Scripting الفعال كما هو الحال بالنسبة للغة VB أو لغة JavaScript.

❖ إمكانية تدعيم الـ Script بلغة HTML.

المكونات الجديدة

تأتى الـ WebSnap مع مضيف لعدد من المكونات الجديدة والتى تتضمن أداة تحفيز الأفعال Dispatcher والمنظم adapter وأداة إنتاج الصفحة والـ Session وقائمة المستخدم.

مكونات أداة تحفيز الأفعال Dispatcher

مكونات أداة تحفيز الأفعال Dispatcher تعالج تلقائيا أنواع مختلفة من الطلبات الخاصة بمحتوى صفحة الويب والمهام الفرعية للفورمة والتى تتم من خلال لغة HTML بالإضافة إلى الطلبات الخاصة بالصور الديناميكية.

المكونات الجديدة والخاصة بأداة تحفيز الأفعال Dispatcher تتضمن الآتى :

❖ المكون TPageDispatcher.

❖ المكون TAdapterDispatcher.

❖ المكون TWebDispatcher. وهذا المكون يمتلك الحدث الجديد OnException

والذى يتم التعامل معه بالصورة التالية

TCustomWebDispatcher.OnException.

مكونات المنظم Adapter Components

المنظمات Adapters تعمل على توفير وسائل لتعريف وسيط أو واجهة استخدام Interface تكون قابلة للبرمجة على حسب قوانين وقواعد العمل وذلك من أجل التطبيق

الذى تتولى إعداده. فعلى سبيل المثال يمكن استخدام المكون TDataSetAdapter لإعداد مكونات Dataset تكون قابلة للبرمجة من خلال لغات إعداد Scripts. المكونات الجديدة الخاصة بالمنظم تتضمن الآتى :

- المكون TAdapter.
- المكون TApplicationAdapter.
- المكون TPagedAdapter.
- المكون TLoginFormAdapter.
- المكون TEndUserAdapter.
- المكون TEndUserSessionAdapter.
- المكون TStringsValueList.
- المكون TDataSetValueList.

مكونات أداة إنتاج صفحة الويب Page Producer Components

أدوات بناء الصفحات تعمل على بناء فورم معقدة وجداول للتعامل مع البيانات كما إنها تستخدم XSL لتكوين صفحة ويب. وهذه المكونات الجديدة تتضمن الآتى :

- المكون TAdapterPageProducer.
- المكون TXSLPageProducer.

مكونات Session

من خلال مكونات Session يمكن متابعة المستخدمين النهائيين للتطبيقات التى تتولى إعدادها. والمكونات الجديدة تتضمن الآتى :

- المكون TSessionsService.

مكونات قائمة المستخدم User List

تعمل مكونات قائمة المستخدم User List على توفير طريقة للوصول لأسماء المستخدمين وكلمات السر الخاص بهم وحقوق الاستخدام أو الوصول الممنوحة لهم. ونود هنا القول بأن المكونات الجديدة فى هذا الصدد عبارة عن الآتى :

- المكون TWebUserList.

**أدوات تصميم السطح بـ WebSnap**

عندما تقوم بإعداد Script لتطبيق معد لكى يعمل بأحد خوادم الويب وذلك من خلال WebSnap فإنك تستطيع بناء صفحات ويب ومشاهدة النتائج فى مرحلة التصميم Design-Time داخل محرر الكود البرمجى Code Editor وبصفحات تحرير الكود البرمجى التالية :

- o HTML Script.
- o HTML Result.
- o Preview
- o XML Tree
- o XSL Tree

الدعم الخاص بلغة XML (الإصدار المتكامل Enterprise فقط)
معالج ربط بيانات XML

هذا المعالج الجديد يقوم بإنشاء كود مكتوب بلغة Delphi وهذا الكود يعمل على توفير طريقة بسيطة للغاية للوصول للقات البيانات المعدة بلغة XML والعمل على تحديثها أيضا. وفى هذا السدد نقول أن الكود البرمجى الذى يتم تكوينه يمثل خاصية طبيعية وبديهى وقائمة على نموذج برمجى يجعل التعامل مع البيانات المعدة بلغة XML كما لو كنا نقوم بالبرمجة من خلال أحد مكونات لغة Delphi.

أدوات ربط البيانات المعدة بلغة XML بالمكونات الخاصة بلغة Delphi تعد بديل للتكويد البرمجى المعقد والذى يكون مطلوبا وضروريا لاستخدام واجهات الاستخدام أو الوسائط Interfaces الخاصة بنموذج الكائن لدى لغة XML (المعروف بـ DOM) والذى يعد اختصارا لـ (Document Object Model) كما أن هذه الأدوات تعتبر مناسبة جدا للتطبيقات التى تتولى مهمة معالجة المستندات المعدة بلغة XML.

برمجة المستندات الممهدة بلغة XML

أدوات ربط البيانات المعدة بلغة XML بالمكونات الخاصة بلغة Delphi تكون معتمدة على المكون الجديد المسمى TXMLDocument وكافة الوسائط أو واجهات الاستخدام الملحقه به وهى XMLDocument وXMLNode. وهذا المكون يعمل على توفير نسخة مبسطة من واجهات الاستخدام أو الوسائط الخاصة بالنموذج DOM الذى أشرنا إليه سلفا وذلك من أجل التعامل مع البيانات المعدة بلغة XML.



كل ما يطلب منك فقط هو أن تقوم بإسقاط (وضع) الكائن TXMLDocument في الفورمة التي تتعامل معها ثم تقوم بتحديد قيمة الخاصية FileName وتجعل الخاصية Active عبارة عن True وفي النهاية سيكون لديك حالة وصول لكافة البيانات التي تتولى إعدادها من خلال لغة XML.

برمجة الكائن DOM للعمل عبر أنظمة التشغيل المختلفة

عند أدنى مستوى في الدعم الخاص ببرمجة المستند المد بلغة XML نجد وحدة تفاعل جديدة يطلق عليها xmldom.pas وهذه الوحدة تعمل على توفير خاصية العمل عبر أنظمة التشغيل المختلفة وذلك لواجهات الاستخدام أو الوسائط Interfaces الخاصة بالبرمجة من خلال المواصفة المعيارية 2 W3C DOM Level. هذا واقتد تم تصميم هذه الوحدة الجديدة بحيث يكون لها بناء هيكل أو بناء معمارى مفتوح مما يؤدي إلى جعل من السهولة بمكان تكامل Interfaces مع النموذج DOM المعتمد على الحلول المعدة بلغة XML.

استخدام XML في تطبيقات قواعد البيانات

هناك مجموعة جديدة من المكونات يمكنك من تكامل المستندات المعدة بلغة XML داخل الهيكل المعماري لتطبيقات قواعد البيانات. ومثل هذه المكونات الجديدة تجعل من الممكن استخدام ملفات التحويل المكونة بواسطة XML Mapper وهي عبارة عن أداة يتم استخدامها في مرحلة التصميم لتعريف وتحديد خرائط Mappings تربط بين أى مستند مصمم بلغة XML وبين حزمة البيانات لأى تطبيق من تطبيقات قواعد البيانات.

المظاهر الجديد لمترجم اللغة Compiler المتغيرات Variants

كافة الروتينات المختصة بمعالجة المتغيرات تم نقلها من وحدة النظام System Unit إلى وحدة جديدة تسمى Variants. وبالرغم من إنه فى إمكانك حتى الان استخدام أنواع المتغيرات بدون اشتغال أو ضم الوحدة الجديدة Variants داخل التطبيقات التي تتولى إعدادها إلا إن المتغيرات الخاصة بالتعامل مع مجموعة الدوال RTL تكون موجودة بالوحدة الجديدة Variants. هذا ولكى تستخدم هذه الدوال فسوف تحتاج لضم الوحدة الجديدة Variants فى جملة الاستخدامات بالتطبيق الذى تتولى إعدادها.



تستطيع الآن تعريف أنواع بيانات خاصة أو مفصلة للمتغيرات. كما أن المتغيرات أصبحت الآن تعمل على تدعيم Int64.



أنواع البيانات الرقمية أو العددية Enumerated

أصبح الآن في الإمكان تخصيص قيمة معينة للعدادات Enumerations كما هو موضح من خلال المثال التالي :

type

TInfo = (iZero, iOne, iTwo, iFour .. 4);

التغييرات التي أجريت على وحدة الثوابت Consts Unit

وحدة الثوابت Consts.pas تم تقسيمها إلى ملفين هما : الملف Consts.pas

والملف RTLConsts.pas.

الموجهات directives الجديدة لدى مترجم اللغة

هناك عدد من الموجهات directives الجديدة تم توثيقها في الدليل الخاص بلغة pascal. ونود هنا القول بأن شركة مايكروسوفت يزداد اعتمادها يوما بعد يوم على أعلام المقدمة header flags الخاصة بـ PFI (اختصار للمصطلح Portable executable والذي يعنى قابلية التنفيذ المتنقلة) وذلك للسماح لأى تطبيق بأن يشير بالتوافق مع خدمات نظام التشغيل OS (Operating System) أو يطلب المزيد من خدمات نظام التشغيل المتقدمة. وفى هذا السدد نقول أن مترجم اللغة بـ Delphi أصبح الآن يعمل على تدعيم اثنين من الموجهات الجديدة والتي تعمل على توفير العديد من الخيارات القوية لجعل التطبيقات المعدة بلغة Delphi تعمل بكفاءة عالية فى أنظمة تشغيل ويندوز NT و ٢٠٠٠. وهذين الموجهين هما :

{ \$SetPEFlags <integer expression> ; }

و

{ \$Set PEFlags - integer expression ; }

هذه الموجهات معدة للمبرمجين المحترفين فقط.





الموجهات الشرطية Conditional directives

لو أنك تكتب تطبيقات لكى تعمل فى بيئات الويندوز المختلفة فى هذه الحالة تستطيع أن تستخدم الرمز MSWINDOWS لتجعل التطبيقات التى تتولى إعدادها لديها القدرة على الشعور بوجود بيئة الويندوز وإلا ستشعر بأنها تعمل فى بيئات تشغيل أخرى مثل Linux.

المثال التالى يوضح لنا كيفية استخدام الرمز السالف الذكر :

```
{IFDEF MSWINDOWS}
    Get Desperate;
{ENDIF}
```

الموجهة \$If

هناك موجهة جديد وهو الموجهة \$If الذى يقوم بتقييم تعبير ثابت كما هو موضح من خلال المثال التالى :

```
{$IF Defined(WIN32) and (SomeConst > 12.0) }
...
{$ENDIF}
```

والجديد فى الموضوع أنه يمكن الآن تقييم أدوات تعريف ثوابت الباسكال من خلال الموجهات \$IF. ونود هنا القول بأنه يمكن اختيار وجود رمز للتعريف الشرطى \$IFDEF وذلك من خلال الدالة Defined() كما إنه يمكن اختبار وجود رمز لأداة تعريف لثابت الباسكال من خلال الدالة Declared().

المثال التالى يوضح لنا ما سبق :

```
{$IF Defined(WIN32) and Declared(MyConst)}
...
{$ENDIF}
```

حقل الضبط \$ALIGN

الآن أصبح لدى حقل الضبط \$ALIGN الخيارات الجديدة التالية :

```
" {$A1}
" {$A2}
" {$A4}
" {$A8}
```

أداة التجميع الأساسية built-in assembler الجديدة

الآن أصبح Delphi 6 مشتملا على أداة تجميع أساسية built-in assembler جديدة تماما تتميز بالآتى :



الموجهات الجديدة VMTOFFSET و DMTINDEX.

تدعيم التوجيهية instruction الجديد لكل من MMX و SIMD و Enhanced MMX و Intel SSE لكل من وحدات المعالجة المركزية CPU التي من الطراز Pentium Pro و PIII و P4 وكذلك التي من الطراز AMD Enhanced 3D الخاص بـ AMD K7.

تقديم الدعم الجديد للبيانات DQ (اختصار للمصطلح Define Quadword).

التحديث الافتراضي الجديد للثوابت القابلة للتحديد

التحديد الافتراضي لموجهة المترجم \$WRITEABLECONST قد تم تغييره من ON إلى OFF. وهذا يعني أنه يجب عليك بشكل صريح تشغيل هذا الموجهة قبل أن تتمكن من التعامل مع القيم التي تنتمي لنوع البيانات Const.

تقديم الدعم لأنواع الملفات المختلفة

من أجل زيادة الدعم الخاص بإمكانية إعداد تطبيقات تكون لديها القدرة على العمل عبر مختلف أنظمة التشغيل نجد أن مترجم اللغة أصبح قادرا على التعامل مع الملفات النصية التي تتبع نمط نظام التشغيل Linux. ومعنى هذا أن الملفات النصية أصبحت تشتمل على سطور تنتهي بالحرف الناظر للأسكى كود 10 بدلا من أن تنتهي بالحرف الناظر للأسكى كود 13 كما هو الحال بالملفات النصية المستخدمة في بيئة الويندوز.

التغييرات التي أجريت على وظيفة التحميل الزائد Overload

العملية التي يقررها المترجم الخاصة بلغة Delphi (والتي تؤدي إلى حدوث تحميل زائد على مترجم اللغة من أجل العثور على أفضل معامل من ضمن مجموعة من المعاملات) قد تم تحديثها في Delphi 6 لتقديم الدعم للحالات التالية والتي كان يعتبرها المترجم قبل ذلك حالات ميثوس منها تماما :

أصبح الآن لدى مترجم Delphi 6 القدرة على تمييز دوال (أو وظائف) التحميل الزائد والتي تشتمل على AnsiString/PChar عن تلك التي تشتمل على المعاملات WideString/WideChar في نفس موضع المعامل.

من الممكن الآن جعل المتغيرات Variants كما لو كانت معاملات في جمل الإعلانات عن دوال التحميل الزائد. ونود هنا القول بأن أي Variant يمكن



اعتباره أكثر عمومية من أى نوع بيانات بسيط.
 عملية تمرير حالات الكائن إلى دوال التحميل الزائد من خلال نسخة من الكائن
 ونسخة من الـ Interface لم تعد تسبب أى مشاكل لترجم اللغة.
 أصبح من الممكن الآن تقبل NIL على أساس أنها قيمة صحيحة لأى من المعاملات
 الخاصة بنوع الـ Interface داخل أى من دوال التحميل الزائد أو الفوقى.

المظاهر الجديد لأدوات التحكم ActiveX والنموذج COM (الإصدار الفنى Professional والإصدار المتكامل Enterprise)

هناك العديد من المظاهر والإمكانيات التى تم إضافتها لكل من أدوات التحكم
 ActiveX والنموذج COM نذكر منها ما يلى.

التسجيل والتركيب لصفات تهيئة النموذج COM

تستطيع الآن تخصيص صفات النموذج COM+ للكائنات COM الجديدة.

المعالج Event Object Wizard

الشكل التالى يوضح لنا المعالج الجديد COM+ Event Wizard :

شكل توضيحي :

وهذا المعالج يسمح لك بأن تنشأ كائنات COM+.

ينبغي عليك حتى الآن إضافة كود برمجى بنفسك لجعل الأحداث تقع
 أو لجعل كود برمجى معين يستجيب للأحداث التى تقع.





تنفيذ الInterfaces الموجودة بالفعل

الشكل التالي يوضح لنا المعالج COM Object Wizard :

شكل توضيحي :

الآن تستطيع أن تستخدم هذا المعالج لتكوين كائن خادم من أجل إعداد Interface افتراضية تكون منتمية لمكتبة الأنواع المسجلة لدى نظام التشغيل الذي تتعامل معه. وفي السابق كان هذا المعالج يقوم دائما بتنفيذ أى Interface يتم إنشاؤه حديثا (نزولا من Unknown). والآن نجد أن المعالج COM Object Wizard يسمح لك أيضا بأن تختار Interface من أى مكتبة من مكتبات الأنواع المسجلة لدى النظام.

بالإضافة لما سبق نجد أن المعالج COM Object Wizard يقوم بإنشاء معلومة مكتبة النوع من أجل القطاع CoClass لكم، يتم تنفيذ هذا الInterface وفي نفس الوقت يتم استخدام قطاع من قطاعات التنفيذ مع عدد من الطرق الهيكلية (بدلا من أن تقوم بذلك بنفسك) لاستكمال عملية التنفيذ.

قطاع التنفيذ يتوارث كل من طريقة التنفيذ Unknown وطريقة التنفيذ IDispatch.



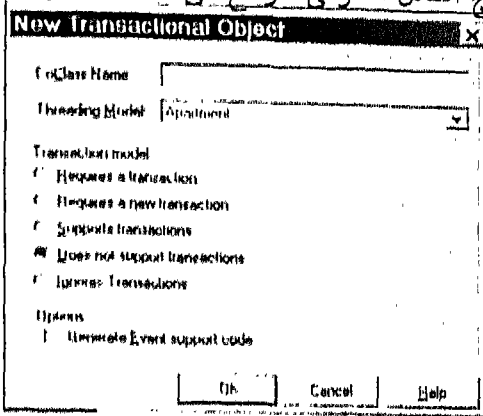
الكائنات المركبة Transactional (كبدل للمعالج MTS)

أصبح في الإمكان الآن إنشاء الكائنات الحركية Transactional Objects باستخدام الإصدار الفني Professional للغة Delphi 6. وفي السابق كان الدعم MTS قاصرا على الإصدار المتكامل Enterprise فقط.



الدعم MTS/COM+ المزود للكائنات الحركية

معالج الكائن MTS تم استبداله بمعالج الكائن الحركي الموضح في الشكل التالي :



شكل توضيحي :

هذا المعالج الجديد لديه القدرة على إنشاء كائنات يمكن استخدامها مع COM+

أو مع MTS.

المظاهر الجديدة لقواعد البيانات

(الإصدار الفني Professional والإصدار المتكامل Enterprise)

يشتمل الإصدار السادس من لغة Delphi على آلية جديدة للوصول للبيانات وهي تعرف بـ dbExpress وهذه الآلية تعمل على توفير وسيلة سريعة للغاية وسهلة للغاية للوصول لخوادم قواعد البيانات SQL وذلك باستخدام إطار هيكلي Framework يجعل من السهولة بمكان كتابة مشغلات قواعد بيانات تنتمي للطائفة Third-party.

مجموعات البيانات Datasets أصبحت الآن تقدم الدعم لنوعين جديدين من الحقول.



لقد تم إضافة عدد المكونات الجديدة إلى Delphi 6 لجعل من الأسهل التعامل مع مجموعات بيانات العملاء Client Datasets وذلك من أجل إعداد تطبيقات قواعد بيانات ذات طبقتين two-tier أو ذات طبقات متعددة Multi-tier.

آلية الوصول للبيانات dbExpress

آلية الوصول للبيانات dbExpress عبارة عن مجموعة من مشغلات قواعد البيانات البسيطة lightweight تعمل على توفير وسيلة سريعة للوصول لخوادم قواعد البيانات



SQL. وبالنسبة لكل قاعدة بيانات تم تدعيمها نجد أن الآلية dbExpress تعمل على توفير مشغل Derive تعمل على ضبط برنامج لخدام معين لإعداد interfaces متجانسة للآلية dbExpress. وعندما تقوم بنشر وتوزيع التطبيق الذى تتولى إعداده تحتاج لأن تضم إما ملف DLL واحد فقط (خاصة بمشغل خادم معين) مع ملفات التطبيق الذى تقوم ببنائه مع اختيار عدد من الملفات الإضافية التى تسمح لك بأن تقوم بتحميل معلومات عن الاتصال مع الخادم المعنى فى أثناء مرحلة التشغيل Run-Time.

كذلك تستطيع نشر وتوزيع التطبيق الذى تتولى إعداده على أساس كونه برنامج تنفيذى EXE مستقل بذاته Standalone.



بناء على الاصدار الذى تستخدمه من لغة Delphi 6 سيكون Delphi مشتملا على المشغلات التالية والتى تخص الآلية dbExpress :

اسم الملف	المشغل
DBEXPINT.DLL	InterBase
DBEXPDB2.DLL	DB2
DBEXPORA.DLL	Oracle
DBEXPMYS.DLL	MySQL.

المكونات الأساسية والمعدة خصيصا من أجل التعامل مع المشغلات الخاصة بالآلية dbExpress عبارة عن الآتى :

- المكون TSQLConnection.
- المكون TSQLDataSet وهو عبارة عن مجموعة بيانات إحادية الإتجاه unidirectional.
- كل من المكون TSQLQuery والمكون TSQLStoredProc والمكون TSQLTable وهى عبارة عن مجموعات بيانات تتصف بإنها إحادية الإتجاه unidirectional وفى نفس الوقت متوافقة مع العناصر الموجودة بالفعل وذلك فى حالة أنك ترغب فى port أى تطبيق.



مجموعات البيانات التى تستخدم الآلية dbExpress يطلق عليها مجموعات بيانات إحادية الإتجاه unidirectional وذلك لأن الآلية dbExpress تقوم بتنفيذ مؤشر إحادى الإتجاه فقط للوصول للسجلات. وفى هذا الصدد نقول إن مجموعات البيانات الإحادية الإتجاه لا تحتفظ بالبيانات بالذاكرة العشوائية RAM من خلال الخاصية buffer مما يجعلهم أسرع وأقل استهلاكاً لمصادر النظام من الأنواع الأخرى من مجموعات البيانات ولكن فى نفس الوقت تؤدي إلى وجود العديد من القيود.

فعلى سبيل المثال لا تستطيع أن تجعل مجموعة بيانات إحادية الإتجاه تقيم اتصال مع شبكة بيانات (مجموعة بيانات ثنائية الإتجاه) وذلك لأنه ليس هناك أى سجلات مخزنة بذاكرة النظام.

العديد من القدرات والإمكانات الخاصة بمجموعة البيانات TDataSet تكون إما غير قابلة للتنفيذ داخل مجموعات البيانات الإحادية الإتجاه أو تجعل تصرفاتهم خارج أى توقعات ممكنة. هذا وبالرغم من القيود المفروضة على مجموعات البيانات الإحادية الإتجاه إلا إنه يمكن اعتبار هذه النوعية من مجموعات البيانات وسيلة قوية للوصول للبيانات. فهي تتميز بالسرعة والبساطة الشديدة سواء عند الاستخدام أو عند النشر والتوزيع.

لكى تقوم بتحرير بيانات من مجموعات بيانات إحادية الإتجاه عليك أن تقيم اتصال بينهم وبين مجموعة بيانات عميلة client dataset. ونود هنا القول بأن الاتصال بين مجموعة البيانات العميلة ومجموعة البيانات الإحادية الإتجاه يتم من خلال مصدر توفير مجموعات البيانات.

أنواع الحقول الجديدة

لقد تم إضافة نوع الحقل الجديد TFMTBCDField إلى Delphi 6. وهذا النوع الجديد يمثل توكيد ثنائى عشرى BCD (اختصار للمصطلح Binary-Coded Decimal) حقيقى فى مقابل نوع الحقل TBCDField الموجود بالفعل والذى يعمل على تحويل القيم BCD لتصبح تابعة للنوع Currency. وأنت تستطيع توصيف نوع الحقل TBCDField على أساس كونه حقل رئيسى لحقل BCD من أى مجموعة بيانات. هذا ومن خلال مجموعات البيانات الخاصة بالآلية dbExpress نجد أنه يتم إنشاء مكونات الحقول الديناميكية التى تنتمى لنوع الحقل TBCDField وذلك عند استخدام نوع الحقل TBCDField قد ينتج عنه فقد فى دقة القيم precision.



فى الواقع يمكن القول بأن ما سبق يمكن أن يمتد لكى يشمل كل من مجموعات البيانات BDE, ADO وإحتمال أن يطول مجموعات البيانات IBX.

بالدعم الخاص بنوع الحقل TBCDField نجد أن الروتينات العامة CurrToBCD و BCDToCurr قد تم نقلهما من الوحدة db إلى الوحدة FMTBCD.

هناك نوع جديد من أنواع الحقول الجديدة المضافه إلى Delphi 6 وهو نوع الحقل TSQLTimeStampField الذى يعمل على تدعيم تمثيل القيم التاريخية والزمنية Date/Time المستخدمة فى المشغلات الخاصة بالآلية dbExpress.

المبائل البنائية لمجموعة بيانات العميل Client Dataset

يشتمل الإصدار السادس من لغة Delphi ثلاثة مجموعات بيانات عملاء جديدة وهى تضم مجموعة بيانات مصدرية وموفر أساسى Built-in Provider. ومجموعات البيانات هذه الغرض الأساسى منها تبسيط عملية استخدام موفر ومجموعة بيانات عميلة للإلتقاط التحديثات بالتطبيقات البسيطة جدا.

فيما يلى مجموعات البيانات العميلة الجديدة :

مجموعة البيانات TBDEClientDataSet.

مجموعة البيانات TSQLClientDataSet.

مجموعة البيانات TIBClientDataSet.

هناك قطاع تأسيس مشترك جديد يسمى TCustomClientDataSet أصبح الآن موجودا لمجموعة البيانات TClientDataset ومجموعات البيانات الثلاثة الجديدة السالفة الذكر.

هذه المكونات مصممة من أجل التطبيقات البسيطة فقط. وعندما يستخدم التطبيق العلاقات التبادلية التفصيلية أو يحتاج للحصول على بيانات من خادم عدة مرات فى هذه الحالة نجد أن مستوى أداء يصبح أفضل خاصة عند استخدام موفر خارجى ومجموعة بيانات عميلة.



مجموعة البيانات TClientDataSet لديها العديد من الخصائص الجديدة

التالية :

الخاصية ConnectionBroker :

هذه الخاصية تسمح لك بأن تضيف طبقة زيادة تعمل على إعادة التوجيه إلى مواصفات أى من مكونات الإتصال. ومثل هذه الخاصية تكون مهمة بصفة خاصة عندما ترغب فى استخدام نوع من أنواع الإتصال فى مرحلة التصميم ونوع آخر من أنواع الإتصال عندما تقوم بنشر وتوزيع التطبيق الذى تتولى إعداده. ولو أن لديك العديد من مجموعات البيانات العميلة داخل أحد التطبيقات وكانت كلها تستخدم نفس الخاصية ConnectionBroker فى هذه الحالة تستطيع تغيير المكون الذى يصل بينهم كلهم وبين خادم التطبيق وذلك عن طريق تغيير خاصية واحدة فقط (الخاصية Connection الخاصة بأداة كسر أو قطع الإتصال) وذلك بدلا من الحاجة إلى تغيير الخاصية RemoteServer لكل مجموعة بيانات عميلة بالتطبيق.

الخاصية DisableStringTrim :

هذه الخاصية تسمح لك بأن تتحكم فى طريقة تعامل مجموعة بيانات العميلة للمسافات الموجودة بالقيم التى يدخلها المستخدم بالحقول.

الخاصية XMLData :

هذه الخاصية تمنحك القدرة على الوصول لحزمة البيانات الخاصة بمجموعة البيانات العميلة علما بأن هذه البيانات معدة بلغة XML.

هناك مجموعة جديدة من المكونات التى تسمح لك بأن تقوم بالتحويل بين مجموعات البيانات العميلة والمستندات المعدة بلغة XML.



لقد تم تحسين وتطوير المكون TUpdateSQL بحيث تستطيع الآن أن تستخدم العديد من كائنات التحديث عند التقاط التحديثات باستخدام مجموعة بيانات عميلة. ونود هنا القول بأن الخاصية DataSet الخاصة بالمكون TUpdateSQL قد تغيرت من كونها تتطلب TBDEDataSet إلى كونها تتطلب TDataSet فقط. هذا وعند استخدام مجموعة بيانات عميلة والعديد من كائنات التحديث فإنه فى هذه الحالة ينبغى عليك تحديد قيمة

هذه الخاصية من خلال المعامل DeltaDS الخاص بأداة معاملة الحدث BeforeUpdateRecord الخاصة بالموفر. وفي هذه الحالة لن يتمكن المكون TUpdateSQL من معرفة اسم قاعدة البيانات والـ session من الخاصية DataSet الخاصة بهذا المكون كما إنه ينبغي عليك أيضا تحديد قيمة كل من الخاصية الجديدة DatabaseName والخاصية الجديدة SessionName.

المكون TUpdateSQL يعمل بنفس الطريقة التي كان يعمل بها في الإصدارات السابقة للغة Delphi وذلك عندما يكون هناك كائن تحديث واحد فقط أو عند استخدام مجموعة البيانات BDE (المقابلة للعمل) للإلتقاط التحديثات التي تتم.



الدعم الخاص بالتطبيقات المتعددة الطبقات multi-tiered

أصبح من المتاح الآن مكونين إتصال جديدين وهما يسمحان لك بأن تعمل بمرونة أكبر مع مجموعات البيانات العملية بالتطبيقات المتعددة الطبقات multi-tiered. وهذه المكونين الجديدين هما :

مكون الإتصال المشترك TSharedConnection :

مكون الإتصال هذا يسمح للتطبيق العميل بأن يعمل على تكوين العديد من الإتصالات بالعديد من Modules البيانات البعيدة الموجودة بتطبيق خادم واحد فقط. هذا ومن خلال استخدام المكونات TSharedConnection يؤدي إلى أن كافة الاتصالات بـ Modules البيانات البعيدة تستخدم اتصال واحد مشترك بتطبيق الخادم مما يسمح لتطبيق الخادم بأن يشعر بهذه الكائنات وهذه الإتصالات ومصادرها أيضا.

مكون الإتصال المحلي TLocalConnection :

هذا المكون يعمل كما لو كان مكون إتصال بالنسبة لموفر البيانات المحليين (في نفس التطبيق على أساس أنه مجموعة بيانات عملية). ومن خلال استخدام مكون الإتصال المحلي TLocalConnection تستطيع أن تجعل استخدام الوسيط IAPPServer يتم

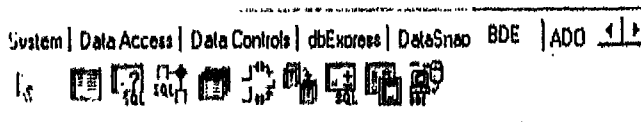
بشكل صريح بالإضافة إلى إمكانية تبسيط عملية Scaling Up التى تتم لاحقا من أجل استخدام موفر بيانات بعيد بأى تطبيق من تطبيقات الخادم.

بالإضافة إلى ما سبق يوجد كائن جديد يسمى TPacketInterceptFactory وهو يجعل من الأسهل اعتراض الرسائل المتبادلة بين تطبيق العميل وتطبيق الخادم وذلك عند استخدام Sockets للاتصال. هذا ومن خلال استخدام الكائن الجديد TPacketInterceptFactory يتم تلقائيا تسجيل كائن الاعتراض ومن ثم تستطيع تخصيص مثل هذا الكائن إلى المكون الذى يعمل كSocket الاتصال وذلك باستخدام قائمة منسدة موجودة فى نافذة أداة فحص الكائن Object Inspector.

التغييرات التى أجريت على باليئة المكونات

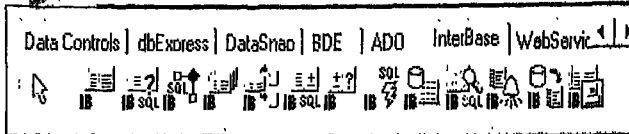
باليئة المكونات قد تم تغييرها بحيث تعكس النمو الهائل فى الخيارات المختلفة والمتعددة التى أصبحت متاحة عند التعامل مع تطبيقات قواعد البيانات وتطبيقات الإنترنت.

باليئة المكونات قد تم إعادة تنظيمها للتأكيد على أن مولد قاعدة البيانات Borland Database Engine الذى أعدته شركة بورلاند (BDE) يعتبر من أقوى مولدات قواعد البيانات الموجودة الآن علما بأن هذا المولد يعتبر بديل للألية الأساسية الوصول للبيانات. وفى هذا الصدد نقول أن المكونات المعتمدة على مولد قاعدة البيانات BDE قد تم نقلها إلى التبويب BDE Page بباليئة المكونات كما هو موضح بالشكل التالى :



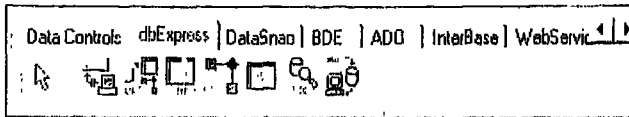
شكل توضيحي :

لقد تم تحديث مجموعة المكونات InterBaseExpress وهذا التحديث اشتمل تحسين وتطوير الدعم المقدم للأحداث المرتبطة بهذه المكونات وكذلك الدعم الخاص بأدوات التكوين InterBase والتى تقوم تلقائيا بتكوين قيم الحقول. ومن ثم فقد تم إضافة صفحة جديدة بباليئة المكونات وهى الصفحة InterBase Admin الموضح فى الشكل التالى :



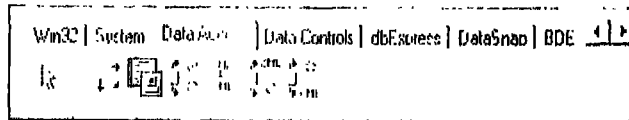
شكل توضيحي :

لقد تم إضافة الصفحة الجديدة dbExpress إلى باليتة المكونات. وهذه الصفحة الجديدة تشتمل على المكونات dbExpress التي سبق التعرض لها في هذا الفصل :



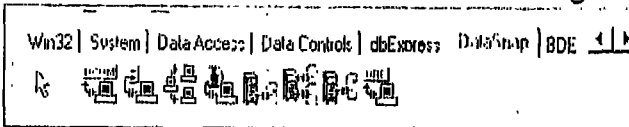
شكل توضيحي :

لقد تم نقل الصفحة Midas كما تم تغيير استخدام المصطلح MIDAS. فكل من مجموعة بيانات العميل ومجموعة بيانات الموفر التي يتم استخدامها ليكونا في الصفحة Midas قد تم نقلهما إلى الصفحة Data Access الموضح في الشكل التالي :



شكل توضيحي :

كل مكونات الإتصال التي يتم استخدامها للاتصال بأي تطبيق من تطبيقات الخادم وأداة تقسيم الكائن البسيط قد تم نقلهما إلى الصفحة الجديدة DataSnap كما هو موضح بالشكل التالي :



شكل توضيحي :

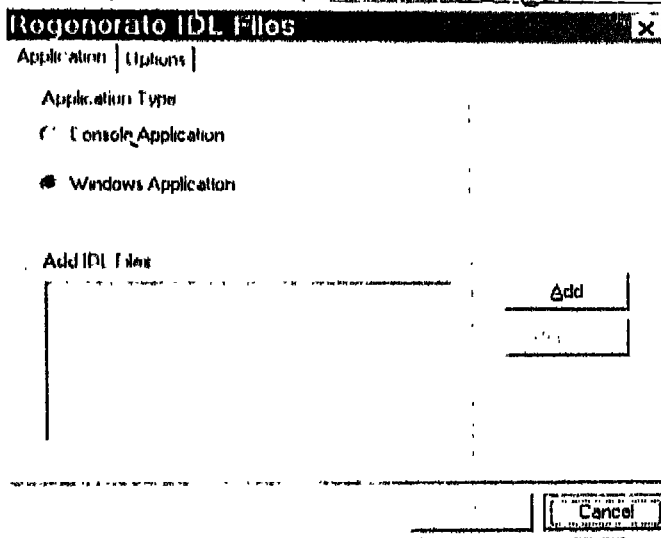
الصفحة الجديدة DataSnap أصبحت الآن مشتملة على المكونات لمستخدمه لإتصال مجموعات بيانات العملاء بأي من تطبيقات الخادم. وهذه المكونات تضم مكونات الإتصال وأداة تقسيم الكائن البسيط والتي كانت موجودة قبل ذلك بالصفحة Midas. بالإضافة لذلك تضم الصفحة DataSnap ثلاثة مكونات جديدة وهي :
المكون TConnectionBroker وهو يعمل كما لو كان مكون وسيط بين مجموعة بيانات العميل وأي مكون من مكونات الإتصال.

- المكون TSharedConnection وهو يتصل بأى تطبيق من تطبيقات الخادم على أساس كونه شريك في العديد من Modules البيانات البعيدة.
- المكون TLocalConnection وهو يمثل الاتصال بموفرى البيانات التى تكون فى نفس التطبيق مثل أى مجموعة من مجموعات بيانات العميل.

المظاهر والإمكانيات الجديد لـ CORBA (الإصدار المتكامل Enterprise فقط)

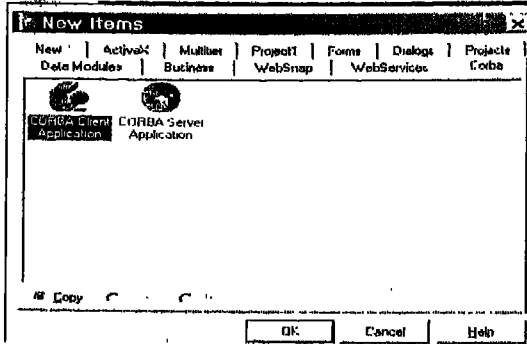
بعض من الإصدارات الخاصة بلغة Delphi تأتى وهى مزودة بالترجم IDL2PAS وذلك لكتابة نوعية خاصة من التطبيقات تعرف بتطبيقات الـ CORBA. ولقد تم تطوير وتحسين المترجم IDL2PAS بحيث أصبحت لديه القدرة على تدعيم عملية كتابة تطبيقات لخوادم الـ CORBA (تكوين كود برمجى هيكلى) بالإضافة إلى كتابة كود برمجى Stub يعمل بتطبيقات العملاء المتصلين بالخوادم الـ CORBA.

هذا وتستطيع الوصول للمترجم IDL2PAS عن طريق فتح القائمة Tools ثم اختيار الأمر Regenerate CORBA IDL Files ليظهر على الشاشة صندوق الحوار Regenerate CORBA IDL Files الموضح فى الشكل التالى :



شكل توضيحي :

أو عن طريق فتح القائمة File ثم اختيار New ثم Other) وبصندوق الحوار New Items انقر بالماوس على التبويب CORBA حتى يظهر على السطح كما هو موضح بالشكل التالى :



شكل توضيحي :

المترجم السالف الذكر يكون مفيداً بالنسبة لعمليات إنشاء كل من تطبيقات العملاء وتطبيقات الخادم بالإضافة إلى صيانة المشاريع CORBA التي سبق إعدادها حيث يمكن أن تجرى تغيير على الملف IDL ثم إنعاش Refresh ملفات المشروع لجعلهم يشعروا بالتغيير الذي أجرته.

من الأشياء التي ننصح بها بشدة هو أن تستخدم المترجم الجديد IDL2PAS الخاص بتطبيقات الـ CORBA وذلك بدلا من استخدام دعم CORBA القديم الذي كان متكاملًا مع الدعم الذي تقدمه لغة Delphi للتطبيقات COM. هذا وعند استخدام المترجم الجديد IDL2PAS في هذه الحالة ينبغي عليك استخدام الوحدة الموجودة بالملف corba.pas والوحدة الموجودة بالملف orbpas30.pas أو تستخدم الوحدة الموجودة بالملف orbpas40.pas للتفاعل مع ORB بدلا من الوحدات القديمة الموجودة بكل من الملف corbaobj.pas والملف orbpas.pas. بالإضافة للوحدات السالفة الذكر فإنه توجد إثنين من الوحدات الجديدة وهما :

الوحدة cosevent.pas :

هذه الوحدة تعمل على تقديم خدمات للأحداث التي تقع للمكونات والوحدة cosnaming.pas.

الوحدة cosnaming.pas :

هذه الوحدة تعمل على تقديم خدمات للتسمية.

التوثيق والمعلومات الخاصة بالدعم المقدم للـ CORBA لم يتم وضعها ضمن التوثيق والمعلومات الخاصة بلغة Delphi. ولكن بدلا من ذلك فقد تم وضع التوثيق والمعلومات الخاصة بالمترجم IDL2PAS بالملف index.htm الموجود بالمجلد IDL2PAS الموجود بدوره بالمجلد Doc.



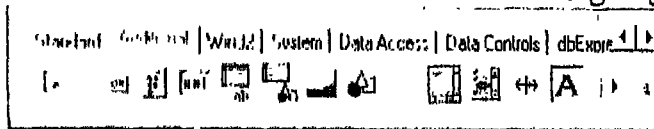


مظاهر وإمكانيات الأفعال Actions الجديدة (الإصدار الفني Professional والإصدار المتكامل Enterprise)

فيما يلي سنتعرف سويا على المظاهر والإمكانيات التي تتمتع بها الأفعال Actions المضافة حديثا للإصدار السادس من لغة Delphi.

حزم الأفعال Action Bands

من السهولة بمكان التعامل مع الأفعال فهي تعمل على تبسيط عملية إعداد وتطوير واجهة الاستخدام للتطبيقات التي تتولى إعدادها وذلك عن طريق استخدام مجموعة من الأدوات الجديدة التي تعرف في مجموعها بحزم الأفعال ActionBands. وأنت تستطيع تنظيم الأفعال والصور كما تستطيع إضافتهم إلى القوائم وشرائط الأدوات التي تتولى تفصيلها بنفسك. وهذه الأدوات يمكن الوصول إليها من خلال الصفحة Additional بالليونة المكونات كما هو موضح بالشكل التالي :



شكل توضيحي :

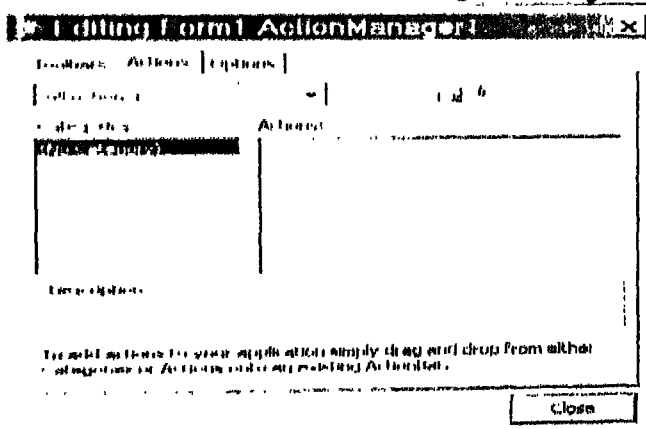
فيما يلي سوف نستعرض سويا مجموعة الأدوات الموجودة بالصفحة

Additional :



الأداة TActionManager :

الشكل التالي يوضح لنا مدير الأفعال Action Manager :



شكل توضيحي :

وهذا المدير يتولى مهمة إدارة قوائم الأفعال وذلك من أجل تنظيم الأفعال سواء كانت مفصلة أو قياسية.



الأداة TActionMainMenuBar و الأداة TActionToolBar :

من خلال الأداة TActionMainMenuBar تستطيع أن تقوم بتفصيل قوائم Menus كما أن الأداة TActionToolBar تعمل على تفصيل شرائط أدوات Toolbars وبكل من هذه القوائم وشرائط الأدوات تستطيع أن تقوم بسحب وإسقاط الأفعال أو الأوامر وذلك من محرر مدير الأفعال. كما تستطيع أيضا سحب وإسقاط مجموعة من الأفعال التي تنتمي لنوع معين (تحصل على كافة الأفعال الموجودة ضمن أى قائمة فرعية) أو تقوم بسحب وإسقاط فعل معين داخل أى قائمة أو شريط أدوات.



الأداة TCustomizeDlg :

من خلال هذه الأداة يتم فتح صندوق حوار التفصيل Customize وهو يعمل بنفس الطريقة التي يعمل بها محرر مدير الأفعال ولكنه مخصص لمستخدمي التطبيق الذى تتولى إعداده وذلك لكى يتمكنوا من تعديل محتويات القوائم وشرائط الأدوات فى أثناء مرحلة تشغيل التطبيق. وفى هذا الصدد نقول إن المكون المتمثل فى الأداة TCustomizeDlg يعمل بالتوافق مع الأداة TActionManager. وأنت تستطيع إما أن تحدد قيمة الخاصية ActionManager للمكون ActionManager أو تخيف فعل قياسي (TCustomizeActionBars) إلى محرر مدير الأفعال وإلى أى من القوائم التى يتم إعدادها من خلال الأداة TActionMainMenuBar. تشتمل لغة Delphi على مجموعة من الأفعال القياسية الجديدة والتي يمكنك إضافتها للقوائم وشرائط الأدوات التى تتولى إعدادها من خلال كل من الأداة TActionMainMenuBar والأداة TActionToolBar. ونود هنا القول بأن قطاعات الأفعال المعرفة مسبقا تضم كل من الآتى :

- o Format
- o File
- o Search
- o Tab
- o List
- o Dialog
- o Internet
- o Tools



تستطيع أيضا إعداد وتهيئة القوائم التى تقوم بتفصيلها لكى تتمكن من إخفاء أو إظهار الأفعال بناء على تتابع استخدام كل منها. وللقيام بذلك عليك أن تستخدم الخاصية HideUnused للعديد من الكائنات TActionClient والتى تعمل على وضع العناصر الأقل استخداما فى القائمة داخل قائمة فرعية يمكن الوصول إليها عن طريق وضع مؤشر الماوس على سهم مزدوج بنهاية القائمة أو شريط الأدوات.

الأفعال القياسية الجديدة

الأفعال القياسية التالية تم إضافتها حديثا إلى VCL :

أفعال التنسيق Format :

- » TRichEditBold
- » TRichEditItalic
- » TRichEditUnderline
- » TRichEditStrikeOut
- » TRichEditBullets
- » TRichEditAlignLeft
- » TRichEditAlignRight
- » TRichEditAlignCenter

الأفعال الخاصة بنظام المساعدة Help :

- » THelpContextAction

الأفعال الخاصة بالتعامل مع الملفات :

- » TFileOpen
- » TFileSaveAs
- » TFilePrintSetup
- » TFileRun
- » TFileExit

الأفعال الخاصة بعملية البحث :

- » TSearchFind
- » TSearchFindFirst
- » TSearchReplace
- » TSearchFindNext

الأفعال الخاصة بالتبويب Tab (أداة التحكم Page) :

- » TPreviousTab
- » TNextTab



الأفعال الخاصة بقوائم العرض List Actions :

- 0 TListControlCopySelection
- 0 TListControlDeleteSelection
- 0 TListControlSelectAll
- 0 TListControlClearSelection
- 0 TListControlMoveSelection
- 0 TStaticListAction
- 0 TVirtualListAction

الأفعال الخاصة بصناديق الحوار Dialog Actions :

- 0 TOpenPicture
- 0 TSavePicture
- 0 TColorSelect
- 0 TFontEdit
- 0 TPrintDlg

الأفعال الخاصة بالإنترنت Internet Actions :

- 0 TBrowseURL
- 0 TDownloadURL
- 0 TSendMail

الأفعال الخاصة بالأدوات Tool Actions :

- 0 TCustomizeActionBars

التحسينات التي أضيفت لقطاعات الأفعال

لقد تم تطوير وتحسين TCustomAction بحيث أصبحت تمتلك الخصائص

الجديد التالية :

- 0 GroupIndex
- 0 SecondaryShortCuts
- 0 HelpKeyword
- 0 HelpType
- 0 AutoCheck

أما TCustomActionList فقد أصبحت تمتلك الآن الخاصية State التي تسمح

لك بأن تقوم بشكل مؤقت بتعطيل كافة الأفعال الموجودة بقائمة العرض List.

المظاهر والإمكانات الجديدة للوحدات VCL الجديدة (كافة الإصدارات)

فيما يلي سوف نستعرض سويًا المظاهر والإمكانات الجديدة للوحدات VCL

الجديدة بالإصدار السادس من لغة Delphi.



المكونات الجديدة

فيما يلي سنتعرف سويا على المكونات الجديدة الخاصة بالوحدات VCL الجديدة.

المكون TLabelEdit :

هذا المكون يعتبر في حد ذاته إضافة للوحدة LabeledEdit.pas وهو يعمل على توضيح كيفية استخدام المكونات الفرعية. وهو عبارة عن أداة تحكم للتعديل Edit تمثلك Label ملحق بها. ونود هنا القول بأن Label يظهر على أساس كونه خاصية لأداة التحكم هذه.

المكون TValueListEditor :

هذا المكون عبارة عن شبكة مفصلة أو خاصية يتم استخدامها لتحرير المكونات TStrings التي تشتمل على أزواج من مفاتيح لوحة المفاتيح والقيم المناظرة لها. وهذا المكون يعمل بنفس الطريقة التي تعمل بها أداة فحص الكائن (Object Inspector).

المكون TComboBoxEx :

هذا المكون عبارة عن أداة تحكم لقائمة منسدلة جديدة وهو يسمح للصور بأن تظهر بجوار العناصر الموجودة بالقائمة المنسدلة.

المكون TColorBox :

هذا المكون عبارة عن أداة تحكم لقائمة منسدلة جديدة تكون مخصصة للتعامل مع الألوان واختيارها.

المظاهر والإمكانيات التي تم تطويرها

أغلب أدوات التحكم التي يتم التعامل معها من خلال بيئة الويندوز أصبحت الآن تعمل على نشر خصائص معالجة الحواف Bevel التالية :

☞ الخاصية BevelEdges.

☞ الخاصية BevelInner.

☞ الخاصية BevelOuter.

☞ الخاصية BevelKind.

☞ الخاصية BevelWidth.

**المكونات الفرعية Subcomponents**

أصبح في إمكان المكونات الآن أن تمتلك مكونات أخرى تعمل على إنشاء مكونات فرعية. فعلى سبيل المثال يمكن لأي مكون أن يكون لديه خاصية تعتبر في حد ذاتها إشارة مرجعية لمكون وهذه الإشارة المرجعية يمكن أن تكون داخلية (لمكون فرعي) أو خارجية (لمكون عادي). هذا ولو أن الإشارة المرجعية الداخلية في هذه الحالة لا يكون المكون الفرعي مملوكا للفورمة نفسها ولكن يكون مملوكا للمكون الموجود في الفورمة. وهذا يعني أنه في إمكان المكونات الآن أن تقوم بنشر خصائص المكون الفرعي بطريقة صحيحة. بالإضافة لما سبق نجد أن أداة فحص الكائن Object Inspector قد تم تعديلها لكي تسمح لك بأن تشاهد خصائص المكونات المرجعية مباشرة inline (مثل الخاصية Font على سبيل المثال).

لكي يتقوم بإنشاء مكون يكون مشتملا على مكون فرعي فإن ذلك يتطلب استدعاء `TComponent.SetSubComponent`.

**خصائص الـ Interface القابلة للنشر**

أصبح من الممكن الآن نشر خصائص الـ Interface (مجموعة من الخصائص تعمل على تحديد نوعية الـ Interface) وذلك في حالة واحدة فقط وهي أن تكون أداة تنفيذ الخاصية عبارة عن مكون streamable. هذا يعني أنك تستطيع الآن مشاهدة الخصائص التي تأخذ Interface في أداة فحص الكائن Object Inspector مما يؤدي إلى توفير قائمة منسدلة تضم المكونات التي تعمل على تدعيم الـ Interface.

الإضافات والتغييرات التي أجريت لوحدات البرمجة

فيما يلي سنتعرف معا على الإضافات والتغييرات التي أجريت لوحدات البرمجة في الإصدار السادس للغة Delphi
الوحدة `CheckList.pas` :

أصبح الآن في إمكان المكون `TCheckListBox` نشر العديد من الخصائص الجديدة بما فيها الخصائص التالية :

الخاصية `AutoComplete`.

الخاصية HeaderColor.

الخاصية HeaderBackgroundColor.

الوحدة Classes.pas :

المكون TList أصبح الآن يمتلك طريقة تخصيص Assign جديدة لا تعمل فقط على إجراء عملية النسخ ولكنها أيضا تسمح بإجراء مجموعة من العمليات الأساسية.

المكون TCollection أصبح الآن يمتلك إنشاء من الطرق المحمية التي يتم استخدامها للسماح بعمل سلات للمكون TCollection وذلك بهدف تحسين وتطوير عملية إضافة ومسح العناصر. وهذين الأسلوبين الجديدين هما الأسلوب Added والأسلوب Deleting. ونود هنا القول بأنه لا يتم تنفيذ كلا الأسلوبين على الفور أي أنهما لا يمتلكان خاصية التنفيذ الافتراضي default implementation. وعلى العموم فليس هناك كل من الحدث OnAdded والحدث OnDeleting لجعل الكائن TCollection صغيرا. ولكن القطاعات المتشعبة من هذا الكائن يمكنها بسهولة إضافة هذه الأحداث. بالإضافة إلى ما سبق نقول أن الكائن TCollection لديه خاصية Owner جديدة لجعل من السهولة بمكان تعريف وتحديد المالك للكائن TCollection.

المكون TStringList أصبح لديه الآن الخاصية الجديدة CaseSensitive والتي تسمح لك بأن تتحكم في تحديد عمليات عرض السلاسل الحرفية (الفرز والترتيب Sorting) والقطايق بين السلاسل الحرفية) التي تتأثر بحالة الحروف (صغير أم كبيرة).

لقد تم نقل المكون TDataModule هنا (Delphi 6) من وحدة الفورم Form وذلك لإزالة الإعتماديات على أدوات التحكم المرئية. وهذا يسمح لك بأن تكتب تطبيقات خادم أصغر حجما حيث إنها لا تتضمن أي واجهة استخدام للتفاعل مع المستخدمين.



المكون TThread أصبح لديه الآن الخاصية الجديدة FatalException التي تعمل على تعريف وتحديد لأي استثناء يمكن أن يؤدي إلى إيقاف دالة Thread من أن تكتمل بشكل طبيعي.

المكون TStream وهو يعمل على إجراء تحميل زائد أو فوقى للدالة SEEK وذلك للسماح للقيم Int64 بأن يتم استخدامها لتعريف وتحديد المواضع. أما بالنسبة للقطاعات المرتبطة بهذا المكون فيمكن أن تتخطى واحد أو التحميل الفوقى الآخر ولكن لا ينبغي أن تتخطى كلاهما.

المكون TInterfacedPersistent وهو عبارة عن قطع تأسيس جديد للكائنات المتبقية دوماً والتي لا تعتبر مكونات ولكن تتولى مهمة تنفيذ Interfaces.

الوحدة ComCtrls.pas :

بهذه الوحدة تم إضافة المكون TTreeView الذي يتولى مهمة إنشاء نقاط التواصل داخل الهيكل الشجري والتي تعرف بـ TreeNodes.

لقد تم الآن تعميم عملية إنشاء نقاط التواصل nodes وإعداد حدث ومن ثم نجد أن مستخدمى الهياكل الشجرية البسيطة لن يحتاجوا بعد الآن إنشاء تفرعات وتشعبات بالهيكل الشجرية وذلك لتخطى قطاع نقطة الاتصال. كذلك فقد تم تغيير AddNode ومن ثم يمكنكم الآن المرور عبر نقطة الاتصال (مهما كان القطاع) التي تود إضافتها للهيكل الشجري.

لقد تم إضافة الحدث OnAddition والذي يقع عند إضافة نقاط اتصال للهيكل الشجري.

لقد تم تبسيط وتنقيح مجموعة الدوال API الخاصة بعملية فرز وترتيب محتوى المشاهد الشجرية TreeViews. ومن ثم تستطيع الآن فرز وترتيب الهياكل الشجرية الفرعية بالإضافة إلى إمكانية ترتيب نقاط الاتصال الموجودة فى قمة الهيكل الشجري. ونود هنا القول بأن كل من القطاع TCustomTreeView والقطاع TTreeNode أصبحا الآن يقدمان تعريفات أكثر تجانس لكل من الأسلوب

AlphaSort والأسلوب CustomSort علما بأن هذه الأساليب قد تم إضافتها إلى TTreeNode.

كافة التغييرات السابقة الذكر لم تؤثر على توافق هذه المكونات مع الإصدارات السابقة للغة Delphi.



لقد تم إضافة كائن الاختيار المتعدد MultiSelect وهو يتمتع بأربعة خصائص وثمانية أساليب.

لقد تم إضافة الخاصية CreateListItems إلى الكون TListView وهي تتطابق مع الخاصية CreateTreeNodes للكون TTreeView.

الآن تظهر نقط تغيير الحجم حول الكون TStatusBar حتى ولو كان شريط الحالة غير مرتبط مباشرة بالفورمة الموجود بها. وطالما أن الركن الأيمن السفلي لشريط الحالة واقعا عند الركن الأيمن السفلي للفورمة في هذه الحالة ستظهر نقط تغيير الحجم على شريط الأدوات.

لقد أصبح لدى الكائن TDateTimePicker الخاصية الجديدة Format وهي تتحكم في طريقة تنسيق القيم التاريخية وذلك باستخدام السلاسل الحرفية القياسية الخاصة بتنسيق التاريخ أو الوقت.

لقد أصبح لدى THeaderControl عددا من الخصائص والأحداث الجديدة لكي يتمكن من تدعيم عملية سحب وإسقاط الأعمدة. ونود هنا القول بأن الخاصية الجديدة HotTrack تسمح بأن يتم التعليم على أقسام الرأس Header وذلك عند تعامل المستخدم مع هذه الأجزاء من خلال الماوس.

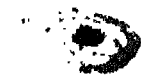
لقد أصبح لدى TToolBar الخاصية Menu الجديدة والتي تعمل على ملء شريط الأدوات بالأيقونات المناظرة للعناصر الموجودة بنظام القوائم. ونود هنا القول بأن هناك عدد من الأحداث الجديدة تقع عندما يقوم المستخدم بتفصيل شريط الأدوات باستخدام أى من صناديق حوار التفصيل.

الوحدة Contrs.pas :

لقد تم إضافة كل من First و Last إلى كل من TObjectList و TComponentList و TClassList والتي تعتبر دوال ذات أنواع خاصة أو مصنعة Typecast.

لقد تحولت الخاصية Push لكل من TStack و TQueue و TObjectStack و TObjectQueue إلى دالة تعطي بكل بساطة العنصر الذي تم دفعه إلى الـ Stack. وعليك أن تفكر فيها على اساس أنها Push/Pek.

هذه الدالة تكون مهمة جدا عند دفع أشياء يتم إنشاؤها في أثناء عملية الدفع.



يمكن استخدام كل من TBucketList و TObjectBucketList لإنشاء جداول بسيطة.

الوحدة Controls.pas :

القطاع TCustomListControl عبارة عن قطاع تأسيس جديد مشترك لأدوات التحكم التي تعمل على تمثيل قائمة تضم مجموعة من العناصر (مثل القوائم المنسدلة Combo Boxes وقوائم العرض List Boxes وقوائم المشاهدة List Views). وهذا القطاع يقدم عدد من الأساليب الجديدة للتعامل مع القوائم Lists التي ستتوارثها بعد ذلك كافة التشعبات المتفرعة من هذا القطاع.

كل من TDragObjectFix و TDragControlObjectFix و TDragDockObjectFix عبارة عن ثلاثة كائنات سحب جديدة وهي تتحرر تلقائيا عند إنتهاء عمليات السحب. وهذه الكائنات تناظر لكل من TDragObject و TDragControlObject و TDragDockObject ولكن الإختلاف الوحيد بينهم يتمثل في أن الكائنات القديمة كانت لا تتحرر عند إنتهاء عملية السحب.

الكائن TControl أصبح لديه الآن أسلوبين جديدين وهما الأسلوب ClientToParent والأسلوب ParentToClient. وهذه الأساليب الجديدة تسمح لك بأن تعد خريطة لنقط الشاشة pixels بحيث يتم ربطهم بواحد من الـ Parents أو الـ Children. وهذه



الأساليب تعمل بطريقة معاكسة جدا لطريقة عمل كل من الأسلوب ClientToScreen والأسلوب ScreenToClient وهما من الأساليب القديمة.

الكائن TWinControl أصبح لديه طريقة جديدة لعمل تحميل فوقى أو زائد للأسلوب PaintTo الذى يتعامل مع نسيج canvas بدلا من HDC.

لقد تم نقل المكون TModalResult من الوحدة Forms.pas إلى الوحدة Controls.pas. بالإضافة لذلك فإن دوال التدعيم التالية قد تم إضافتها إلى الوحدة Controls.pas:

```
function IsPositiveResult(const AModalResult: TModalResult): Boolean;
function IsNegativeResult(const AModalResult: TModalResult): Boolean;
function IsAbortResult(const AModalResult: TModalResult): Boolean;
function IsAnAllResult(const AModalResult: TModalResult): Boolean;
function StripAllFromResult(const AModalResult: TModalResult):
TModalResult;
```

الوحدة DbCtrls.pas :

كل من الكائن TDBLookupComboBox و TDBLookupListBox أصبح لديه الخاصية الجديدة NullValueKey والتي تسمح للمستخدمين بأن يخصصوا قيم فارغة NULL لهذه الكائنات.

أصبح الآن لدى الكائن TDBCComboBox كل من الخاصية الجديدة AutoComplete والخاصية AutoDropDown.

كذلك فإن الكائن TDBListBox أصبح لديه الخاصية الجديدة AutoComplete أيضا.

الكائن TOpenDialog أصبح يمتلك الآن الخاصية OptionsEx التى تعمل على تطوير وتوسيع قدرتك للتحكم فى كل من صندوق حوار الفتح Open وصندوق حوار الحفظ Save.

الوحدة ExtCtrls.pas :

لقد تم إضافة الخاصية Proportional إلى أداة التحكم TImage. وهذه الخاصية تبقى على النسبة بين طول وعرض الصورة ثابتة بغض النظر عن الحجم الحالى . لأداة التحكم TImage.

الوحدة Forms.pas :



كل من TApplicationEvents و TApplication أصبح لديهما الحدث الجديد OnSettingChange الذى يسمح لك بأن تستجيب للتغييرات التى تجرى على القيم التحديدية الأساسية لنظام التشغيل.

لقد أصبح الآن TForm يعمل على تدعيم الفورم ذات الطبقات Layered form وذلك من خلال مجموعة الخصائص الجديدة AlphaBlend و AlphaBlendValue و TransparentColor و TransparentColorValue.

الكائن TScreen لدية مجموعة من الخصائص الجديدة يتم استخدامها للحصول على منطقة عمل بسطح المكتب (هذه الخصائص هى WorkAreaRect و WorkAreaTop و WorkAreaLeft و WorkAreaHeight و WorkAreaWidth). بالإضافة إلى أنه يمتلك مجموعة من الأساليب الجديدة التى تسمح لك بأن تحدد أفضل موضع لمنطقة العمل بسطح المكتب.

يعمل الكائن TMonitor على تطوير وتقوية الدعم الخاصة بإمكانية التعامل مع عدة شاشات فى نفس الوقت وذلك عن طريق تحديد أى من الشاشات تعتبر الشاشة الأساسية أو الرئيسية. كما أن هذا الكائن يمتلك كل من الخاصية WorkAreaRect والخاصية BoundsRect.

لقد تم إضافة الدعم AutoDragDocking لهذه الوحدة. وهذا الدعم يسمح لك بأن تغلق خاصية الرسو التلقائى auto-docking للتطبيق الذى تتولى إعداده. كذلك فإنه تم إضافة flag إلى صندوق حوار الخيارات Options الخاص بلغة Delphi مما يسمح لك بأن تحدد قيمة هذه الخاصية.

لقد تم نقل TModalResult إلى الوحدة Controls.pas.

الوحدة Graphics.pas :

لقد تم إضافة كل من TFontRecall و TPenRecall و TBrushRecall إلى هذه الوحدة. وهذه المكونات تسمح بأن يتم سريعا حفظ واستعادة الفوننتات والأقلام pens والفرش. وهذه المكونات متشعبة من TRecall (التابع لقطاعات الفورم) الذى يعمل مع القطاعات TPersistent بصفة عامة.

لقد تم تخزين ألوان النظام وذلك لجعل من السهولة بمكان العثور على الألوان المطلوبة.

لقد تم إضافة الألوان الأربعة الجديدة التالية إلى الألوان الستة عشر القياسية :



- o clMoneyGreen.
- o clSkyBlue.
- o clCream.
- o clMedGray.

الوحدة ImgList.pas :

لقد تم إضافة أدوات للتحميل الفوقى أو الزائد لكل من الأسلوب Draw والأسلوب DrawOverlay والأسلوب GetIcon مما يسمح لك بأن تتخطى القيم التحديدية للخصائص الخاصة بقائمة عرض الصور Image List.

الوحدة IniFiles.pas :

الآن أصبحت الملفات التى لها الإمتداد .ini تعمل على تدعيم عمليات كتابة وقراءة بيانات ثنائية Binary وذلك باستخدام stream.

الآن يسمح لك المكون TMemIniFile بأن تتحكم فيما إذا كانت السلاسل الحرفية سيتم معالجتها بطريقة تأخذ فى الاعتبار حالة الحروف (كبيرة أو صغيرة) أم لا.

يعتبر المكون THashedStringList فرع جديد من القطاعات TStrings وهو يستخدم جدول داخلى بسيط وذلك بهدف تطوير وتحسين مستوى أداء التطبيق.

الوحدة Masks.pas :

بهذه الوحدة نجد أن كل من EditMask وText يستخدمان الآن أنواع خاصة أو مفصلة ومن ثم أصبحت خصائص التحرير الخاص بهما أكثر إفادة.

الوحدة Menus.pas :

المكون TMenuItem لديه الآن الخاصية الجديدة AutiCheck التى تحدد إمكانية مراجعة أو عدم مراجعة العناصر الموجودة بنظام القوائم تلقائيا وذلك عندما يتعامل المستخدم مع هذه العناصر بالتطبيق.

الوحدة Registry.pas :

أصبح المكون TRegistry الآن يعمل على تدعيم كل من قراءة وكتابة البيانات الثنائية باستخدام Stream.

الوحدة StdCtrls.pas :



- ❖ لقد تم إضافة كل من الحدث OnCloseUp والحدث OnSelect إلى المكون TCustomComboBox (ومن ثم فقد تم إضافتهما أيضا إلى المكون TComboBox).
- ❖ تتم إستثارة الحدث OnCloseUp عند إغلاق القائمة المنسدلة الخاصة بالكائن Combo Box (عليك أن تفكر في ذلك على أساس أنه معاكس للحدث OnDropDown).
- ❖ بالنسبة للحدث OnSelect فتتم استثارتة عند اختيار أى عنصر من القائمة المنسدلة للكائن Combo box (أو عندما يحدث تغيير لمحتوى القائمة المنسدلة عن طريق التحويل في هذا المحتوى لأعلى أو لأسفل).

القوائم المنسدلة Combo Boxes أصبحت الآن تمتلك الخاصية AutoComplete التى تكون true فى بداية التعامل معها.



- ❖ الآن يعمل TListBox على تدعيم نوعين جديدين من الأنماط وهما النمط IbVirtual والنمط IbVirtualOwnerDraw. وهذه الأنماط تكون مخصصة لقوائم العرض التصورية Virtual List boxes والتى لا تحتفظ بأى عناصر داخلها. ولكن بدلا من ذلك فإنك تشير إلى عدد العناصر الموجودة بهذه النوعية الخاصة من قوائم العرض عن طريق تخصيص قيمة للخاصية Count وبعد ذلك تقوم بإمدادهم بالعناصر (وبالكائنات المرتبطة بهم) وذلك باستخدام مجموعة الأحداث الجديدة OnData و OnDataFind و OnDataObject.

الوحدة TypInfo.pas :

- ❖ أصبح من الممكن الآن استدعاء الدالة GetPropInfo مع أى كائن ليس لديه أى معلومات RTTI. وفى مثل هذه الحالة فإننا نحصل من الدالة على nil.
- ❖ لقد تم إضافة المكون FreeAndNilProperties إلى هذه الوحدة. وهذا المكون سيأخذ أى كائن RTTI-enabled object ثم يقوم بتفريغ الخصائص الخاصة به مع ملاحظة أنه سيتم أيضا تفريغ أى كائنات مشار إليها مرجعيا من خلال هذا الكائن.

الوحدات RTL الجديدة والمظاهر الجديدة الخاصة بها (كافة الإصدارات)

بعض الدوال قد تم نقلها من وحدات أخرى إلى وحدة النظام فى حين أن العديد من دوال النظام قد تم نقلها إلى الأخرى إلى وحدة المتغيرات الجديدة. بالإضافة إلى ذلك فقد تم أيضا إضافة العديد من إصدارات التحميل الفوقى الجديدة للدوال RTL وذلك من أجل تدعيم المعاملات WideString بالإضافة إلى المعاملات AnsiString (SysUnits) مثل المعامل Trim والمعامل WideFormat.

الوحدة Variants.pas :

هذه الوحدة الجديدة تشتمل على العديد من الخدمات للتعامل مع المتغيرات. والكثير من هذا الكود البرمجى قد تم نقله من الوحدة System ومن ثم الآن لو أن الكود البرمجى الذى تعدده يستخدم متغيرات فى هذه الحالة ينبغى عليك إضافة Variants إلى الجملة Uses. كذلك فإن هذه الوحدة تشتمل أيضا على عدد من الروتينات التى تعمل على تدعيم استخدام Variants وفى نفس الوقت تدعم أنواع المتغيرات المفصلة أو الخاصة Custom.

لكى يتم تدعيم إمكانية إعداد تطبيقات تكون قادرة على العمل عبر العديد من أنظمة التشغيل المختلفة نجد أن الخدمات الموجودة بالوحدة Variants لم تعد تجرى استدعاءات مباشرة داخل دوال الويندوز API. ولكن بدلا من ذلك هناك وحدة جديدة تسمى VarUtils.pas تشتمل على روتينات منخفضة المستوى low-level تعمل على توفير نظام تأسيس طبيعى من أجل الروتينات الموجودة بالوحدة Variants.pas. وأنت تحتاج لأن تتوخى الحرص عند استخدام الكود التوليدي Generic code بالوحدة VarUtils.pas وذلك بسبب أنه لا يشبه تماما كود الويندوز كما أنك لو قمت بالخلط بين استدعاءات الويندوز والكود التوليدي فى هذه الحالة ستقع فى مشاكل أنت فى غنى عنها.

الوحدة ConvUtils.pas :

هذه الوحدة عبارة عن مجموعة من الروتينات الخاصة بعملية التحويل بين وحدات القياس التى تم إضافتها.



الوحدة StdConv.pas :

هذه الوحدة عبارة عن مجموعة من المتغيرات العامة Global التي يتم استخدامها مع الروتينات الموجودة بالوحدة ConvUtils.pas.

الوحدة DateUtils.pas :

هذه الوحدة عبارة عن مجموعة من دوال التاريخ والوقت تم إضافتها حديثا للغة Delphi 6.

الوحدة StrUtils.pas :

هذه الوحدة الجديدة تشتمل على دوال حرفية بالإضافة إلى الدوال الموجودة بالوحدة SysUtils.pas.

الوحدة FMTBCD.pas :

هذه الوحدة الجديدة تشتمل على خدمات للتعامل مع القيم BCD (اختصار للمصطلح Binary-Coded Decimal).

الوحدة Math.pas :

لقد تم إضافة Const إلى مجموعة المعاملات القابلة للامتداد من أجل الإسراع من إجراء العمليات الرياضية. كذلك هناك العديد من الثوابت والدوال بهذه الوحدة.

الوحدة System.pas :

الوحدة System لديها العديد من الروتينات الجديدة والتي أغلبها تعمل على تدعيم إمكانية إعداد وبرمجة تطبيقات تكون قادرة على العمل عبر العديد من أنظمة التشغيل المختلفة كما تدعم أيضا إمكانية التحويل بين الأنظمة المختلفة لتكويد الحرفيات المستخدمه بكل من بيئة الويندوز وLinux.

الروتينات التي تعمل على تدعيم المتغيرات Variants قد تم نقلها من الوحدة System.pas إلى الوحدة Variants الجديدة.



لقد تم إضافة IInterface إلى الوحدة System.pas لكى يتم استخدامها مع الـ Interfaces التى ليست COM.

الوحدة SysUtils.pas :

لقد تم تحديث الوحدة SysUtils بحيث تعمل على تدعيم مظاهر وإمكانيات الـ Cross-Platform الخاصة بـ CLX. وهذا التحديث يتمثل فى أن بعض الخدمات التى كان يتم توفيرها فى السابق من خلال دوال الـ WinAPI الأساسية أصبح الآن يتم توفيرها من خلال الروتينات الموجودة بالوحدة SysUtils. ولكن على كل حال العديد من هذه الروتينات يظهر فى المناطق المخصصة للتعامل مع نظام التشغيل Linux من ملف الكود البرمجى الأصلى وكما لا يتم ترجمتها إلى النظام الثنائى الذى تعتمد عليه بيئة الـ ويندوز.

تدعيم أنواع المتغيرات الخاصة (كافة الإصدارات)

تستطيع الآن تعريف أنواع بيانات خاصة وتخصيصها لأى متغيرات. وهذه الإمكانية تعمل على تقديم معامل تشغيل التحميل الفوقى فى أثناء تخصيص النوع الخاص إلى المتغير. هذا وإنشاء نوع متغير جديد عليك أن تكون فى مكون منحدر من القطاع أو TCustomVariantType (أو واحد من الكائنات المنحدرة منه TInvokeableVariantType أو TPublishableVariantType) ثم تعلن عن حالة instantiate لنوع المتغير الجديد.

فيما يلى سناهد معا إثنين من الوحدات الجديدة التى تقدم لنا أمثلة للمتغيرات الخاصة :

الوحدة VarCmplx تقوم بتنفيذ نوع متغير خاص للأرقام المركبة. ونود هنا القول بأن نوع المتغير يعمل على تدعيم المعالجة المباشرة باستخدام معاملات التشغيل للجمع والطرح والضرب والقسمة (ولكن ليست قسمة القيم العددية الصحيحة Integer) والنegation. وفى هذا الصدد نقول إن نوع المتغير يعمل على تدعيم خصائص وهى : Real و Imaginary و Radius و Theta و FixedTheta. كذلك يمكن تحويل نوع المتغير لتكوين أنواع من البيانات العددية الصحيحة Integer Types وأنواع بيانات عددية عشرية floating point وأنواع بيانات حرفية



string وأنواع بيانات تاريخية وزمنية TDateTime وكذلك أنواع بيانات منطقية Boolean. بالإضافة لذلك نجد أن الوحدة VarCmplx تقوم بتنفيذ عدد من الدوال العامة من أجل إجراء عمليات على المتغيرات المركبة complex.

● الوحدة VarConv تعمل على تنفيذ إمكانية إعداد أنواع متغيرات خاصة وذلك من أجل القياسات ووحدات القياس مثل المستخدم في الوحدة ConvUtils. ونود هنا القول بأن نوع المتغير الخاص Convert يعمل على تدعيم عمليات الجمع والطرح والضرب والقسمة التي تجرى بين نوع المتغير الخاص Convert والأرقام أو بين اثنين من نوع المتغير الخاص Convert (فيما عدا أنك لا تستطيع ضرب اثنين من أنواع المتغيرات الخاصة Convert في بعضهما البعض في حال أنهما لا يستخدمان نفس الوحدات). نوع المتغير Convert يقوم تلقائياً بضبط الوحدات عندما تقوم بأداء العمليات السالفة الذكر. وأنت تستطيع تحويل cast المتغيرات التي من النوع Convert بحيث تصبح OleStr و String و Double. كما إنها تعمل أيضاً على تدعيم الخصائص Value و Type و TypeName و Family و FamilyName بالإضافة للخاصية As<Unit> وذلك للحصول على كل من قيمة عددية ونوع وحدة القياس واسمها ومجموعة وحدات القياس التي تنتمي إليها وحدة القياس المستخدمة حالياً. بالإضافة إلى الحصول على القيمة الناتجة من تحويلها إلى وحدة قياس أخرى في نفس عائلة وحدات القياس.

إمكانية إعداد تطبيقات تعمل عبر مختلف نظم التشغيل (الإصدار الفني Professional والإصدار المتكامل Enterprise)

تأتي لغة Delphi 6 وهي مزودة بقطاع مكتبي يعرف بـ CLX (اختصار للمصطلح Borland Component Library for Cross-Platform). وهذا القطاع المكتبي يشبه القطاع VCL الذي يمكن تشغيله في كل من بيئة الويندوز وبيئة Linux. وفي هذا الصدد نقول إن الكائنات التي تنتمي للقطاع CLX قد تم تسميتها بنفس أسماء الكائنات التي تنتمي للقطاع VCL كما إن كلا النوعين من الكائنات لديهما العديد من الخصائص والأساليب والأحداث المتشابهة. وأنت تستطيع استخدام القطاع المكتبي CLX مع Delphi 6 وذلك من أجل إعداد تطبيقات يمكن ترجمتها من خلال كل من بيئة الويندوز وبيئة Linux.

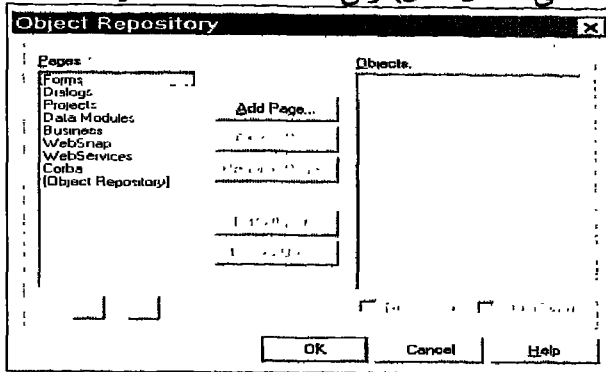


الاختلافات فى القطاع المكتبى CLX

هناك إختلافات نتيجة لنظام التشغيل والمظاهر والإمكانيات التى ترتبط بالتكنولوجيات التى تكون مقصورة على بيئة الويندوز فقط :

هناك تشابه إلى حد كبير بين القطاع CLX والقطاع VCL بالرغم من أن أداة التحكم TWidgetControl بالقطاع CLX تحل محل أداة التحكم TWinControl بالقطاع VCL. ومن ثم فإن المكونات الأخرى مثل المكون TScrollingWidget يكون لها نفس الأسماء بكلا القطاعين.

بيئة التطوير المتكاملة IDE تستخدم مجموعة فرعية أصغر من الكائنات فى النافذة Object Repository (الموضحة فى الشكل التالى) وفى صفحات بالبيئة المكونات :



شكل توضيحي :

كافة الأكواد البرمجية المسؤولة عن تأمين المتغيرات والتى كانت فى الوحدة System أصبحت الآن فى الوحدات الجديدتين : Variants.pas و VarUtils.pas.

تستطيع أن تقوم بتعريف أنواع بيانات خاصة مفصلة Custom للمتغيرات التى تتعامل معها. وهذه الإمكانية تؤدى إلى التقديم لمعامل تشغيل للتحميل الفوقى فى أثناء تخصيص نوع البيانات الخاص أو المفصل للمتغير. ونود هنا القول بأنه لإنشاء نوع بيانات جديد لأى متغير عليك أن تعد تشعب جديد من القطاع ثم تستخدم TCustomVariant ثم تعد حالة ابتدائية instantiate لنوع البيانات الجديد الذى سيتم تخصيصه للمتغير.



يمكن استخدام أنماط التطبيقات المعدة للعمل عبر مختلف أنظمة التشغيل بالإضافة إلى خصائص الكائن OwnerDraw. وأنت تستطيع استخدام الخاصية Style للكائن TApplication وذلك لتوصيف وتحديد مظهر وسلوك العناصر الرسومية بهذه النوعية من التطبيقات.

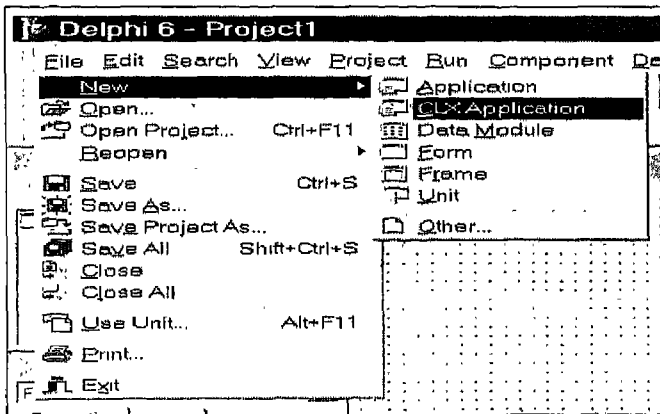
أغلب السلاسل الحرفية Strings الخاصة بأدوات التحكم الموجودة بالقطاع CLX تعتبر سلاسل حرفية متسعة Wide في حين أن أغلب السلاسل الحرفية الخاصة بأدوات التحكم الموجودة بالقطاع VCL عبارة عن سلاسل حرفية من النوع ANSI. ونود هنا القول بأن إجراء عملية تحويل النوع Typecasting لسلسلة حرفية متسعة لتصبح PChar أو إجراؤها على النوع String ليصبح PWideChar لن تنجح وستتسبب في جعل مترجم اللغة يعرض رسالة التحذير Suspicious typecast.

نظام التشغيل Linux لا يستخدم أحقية استخدام registry لتخزين معلومات للتهيئة. ولكن بدلا من ذلك فإنك تستخدم ملفات التهيئة النصية و متغيرات بيئة العمل للقيام بهذه المهمة.

إنشاء تطبيق يعمل بمختلف أنظمة التشغيل cross-platform application

لكي تنشأ تطبيق يعمل بمختلف أنظمة التشغيل اتبع الخطوات التالية :

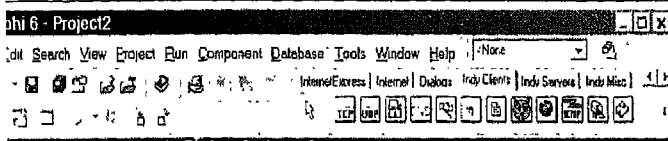
- (١) افتح القائمة File ثم اختر منها New ثم CLX Application كما هو موضح بالشكل التالي :



شكل توضيحي :



(٢) ستجد أن بالليته المكونات تتغير ديناميكيا لكى تعرض الكائنات التى تكون متاحة للاستخدام فى التطبيقات CLX التى ستعمل بيئة الويندوز كما هو موضح بالشكل التالى :



شكل توضيحي :

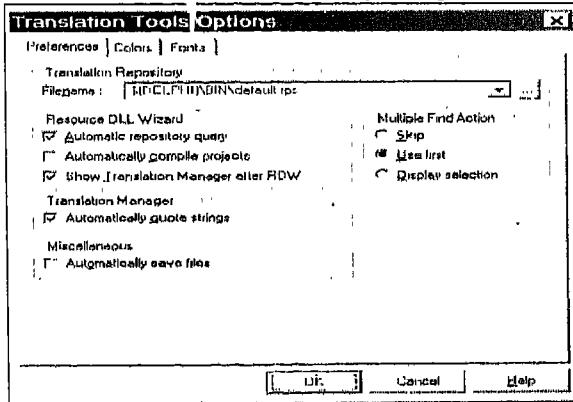
هناك بعض المظاهر المعينة من مظاهر الويندوز لا يتم توفيرها للتطبيقات التى ستعمل بنظام التشغيل Linux.



أدوات الترجمة والمطورة (الإصدار المتكامل Enterprise)

مجموعة أدوات الترجمة Translation Tools والتى تضم كل من مدير الترجمة Translation Manager والمعالج Resource DLL Wizard بالإضافة إلى مستودع الترجمة Translation Repository قد أصبح اسمها بيئة الترجمة المتكاملة ITE (Integrated Translation Environment). والآن أصبح مدير الترجمة مستقلا بذاته ويمكن تشغيله بصفة مستقلة مما يمكن استخدامه خارج بيئة التطوير المتكاملة IDE. ونود هنا القول بأنه عند استخدام مدير الترجمة من خارج بيئة التطوير المتكاملة IDE فى هذه الحالة يطلق عليه مدير الترجمة الخارجى ETM (External Translation Manager) (فى هذه الحالة يتم تشغيله من الملف etm60.exe) كما يمكن إرساله إلى المترجمات بدون حاجتهم لأن يتم تركيب لغة Delphi.

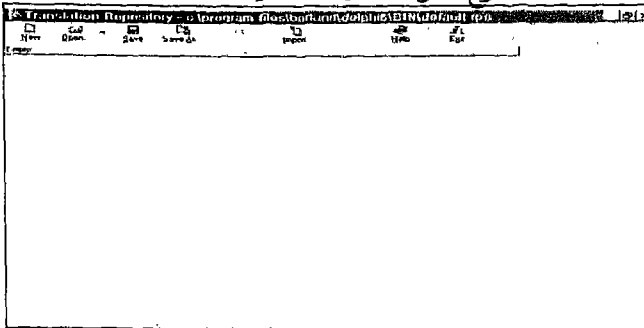
الشكل التالى يوضح لنا صندوق حوار خيارات أدوات الترجمة Translation Tools Options



شكل توضيحي :

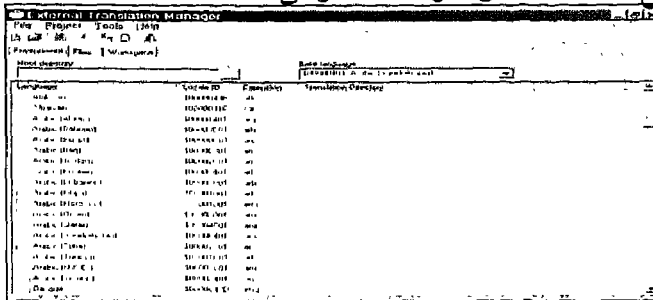


أما الشكل التالي فيقدم لنا نافذة مستدوع الترجمة Translation Repository :



شكل توضيحي :

أما الشكل التالي فيوضح لنا نافذة مدير الترجمة الخارجي ETM :



شكل توضيحي :

صندوق الحوار الخاص بكل من مدير الترجمة TM ومدير الترجمة الخارجي

ETM أصبحت تتمتع بالخصائص الوظيفية الجديدة التالية :

• أداة مشاهدة الفورمة التي منحت المستخدمين إمكانية تغيير حجم الفورمة في أثناء إجراء عمليات الترجمة.

• تم تقسيم نافذة مدير الترجمة الخارجي ETM إلى ثلاثة تبويبات وهي تبويب بيئة العمل Environment وتبويب الملفات Files وتبويب منطقة العمل Workspace. وهذه التبويبات تساعد في تنظيم المشروع الذي تتولى إعداده.

• تم إضافة كل من القائمة File والقائمة Project والقائمة Tools إلى نافذة مدير الترجمة الخارجي ETM كما تم إضافة شريط أدوات يضم أيقونات تناظر الأوامر الموجودة بهذه القوائم. وهذه الأوامر تعتمد بشدة على ما إذا كنت تقوم بتشغيل مدير الترجمة TM داخليا أم خارجيا.



التغييرات التي أجريت على طريقة نشر وتوزيع التطبيقات (كافة الإصدارات)

في الإصدار الخامس من لغة Delphi كان المستخدمون يقومون بنشر الحزم البرمجية التشغيلية runtime packages عن طريق توزيع الحزم المناسبة (الموجودة بالملفات ذات الامتداد BPL). مع التطبيقات الخاصة بهم وذلك باستخدام InstallShield من أجل أن يتم تلقائياً ضم المكتبات الضرورية.

ولكن الحال تغير مع الإصدار السادس من لغة Delphi فهذا الأصدار يأتي وهو مزود بالإصدار الثالث من InstallShield Express المبني على تكنولوجيا التركيب Windows Installer (التي ابتكرتها شركة مايكروسوفت) (وهي تعرف بـ MSI) كما يستخدم Modules الدمج وذلك لنشر التطبيقات التي يتم إعدادها. ونود هنا القول بأن Modules الدمج عبارة عن مكونات MSI تشتمل على الملفات والمخطط المنطقي الضروري لتركيبة مكتبات مرحلة التشغيل Run-Time Libraries.

المكتبات المتاحة لدى لغة Delphi تمتلك الإعتمادات البيئية مع Modules الدمج المتاحة أيضاً لدى لغة Delphi والمصممة بحيث تتفاعل مع هذه المكتبات بكفاءة تامة.



زيادة كافة نظام المساعدة (كافة الإصدارات)

لغة Delphi الآن تسمح لك بأن تمرر طلبات المساعدة إلى أدوات مشاهدة محتويات نظام المساعدة التي تختلف عن الأدوات القياسية التي توفرها بيئة الويندوز لمشاهدة مواضيع المساعدة مما يمكنك من كتابة وإعداد أنظمة المساعدة للتطبيقات التي تتولى إعدادها سواء التي ستعمل ببيئة الويندوز أو بنظام التشغيل Linux.

فيما يلي سنستعرض سوياً المظاهر والإمكانيات الجديدة التي أضيفت حديثاً لنظام المساعدة :

• واجهات الاستخدام الجديدة New Interfaces والتي تعد وسيلة الاتصال بين التطبيق الذي تتولى إعدادها وأدوات مشاهدة محتوى نظام المساعدة. وهذه الواجهات الجديدة معرفة في الملف HelpIntfs.pas وهي تضم كل من الآتي :

- 0 IExtendedViewer
- 0 ISpecialHelpViewer



- o IHelpManager
- o IHelpSystem
- o IHelpSelector
- o ICustomHelpViewer

• مدير المساعدة Help Manager والذي يحتفظ بقائمة تضم أدوات المشاهدة المسجلة والطلبات التي تم تمريرها إلى كل منهم.

• يعمل القطاع VCL على توفير إمكانية لتنفيذ ICustomHelpViewer المصممة للتحدث مع WinHelp.

مصممى ومطورى المكونات التى تنتمى للطائفة Third-Party لديهم الرغبة فى توفير نظام مساعدة باستخدام مولد المساعدة القياسى الذى تدعمه بيئة الويندوز. ولتحقيق هذه الأمنية يجب عليهم الآن توفير قائمة بكافة الـ ALinks المستخدمة فى ملف المساعدة المترجم Compiled. وهذه القائمة يجب أن تكون مخزنة فى ملف له الامتداد ALS. كما يجب وضع نسخة من هذا الملف داخل الفهرس الموصف بواسطة مدخل التسجيل للغة Delphi. هذا ويمكن إنشاء القوائم التى تضم ALinks من خلال ملفات المساعدة المترجمة وذلك باستخدام الخاصية Report فى ورشة عمل المترجم الخاص بنظام المساعدة (والمركب داخل مجلد الأدوات بالفهرس Help الموجود بدوره فى المجلد الذى تم تركيب لغة Delphi به).



الفصل الثانى

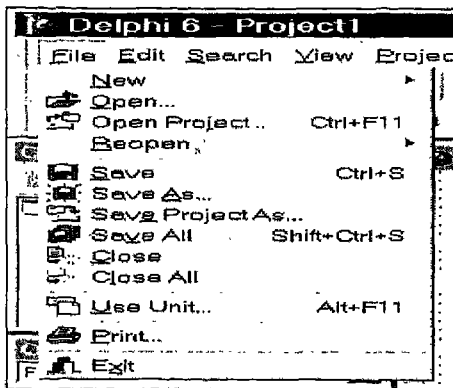
أدوات العمل
بيئة التطوير المتكاملة IDE
لغة Delphi 6



القائمة File

يتم استخدام القائمة File لفتح وحفظ وعلق وطباعة المشاريع سواء كانت جديدة أو كانت موجودة بالفعل كما يمكن استخدامها أيضا لإضافة فورم ووحدات جديدة لفتح مشروع جديد.

الشكل التالى يوضح لنا محتويات هذه القائمة :

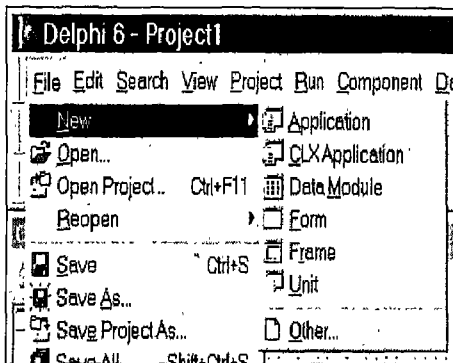


شكل توضيحي

الأمر New بالقائمة File

يعمل هذا الأمر على عرض القائمة الفرعية New كما يعمل أيضا على فتح صندوق الحوار New Items والذي يشتمل على الكائنات المخزنة فى مخزن الكائنات Object Repository كما يشتمل أيضا على مجموعة من المعالجات wizards المستخدمة فى إنشاء كائنات جديد.

الشكل التالى يوضح لنا محتويات القائمة الفرعية New :



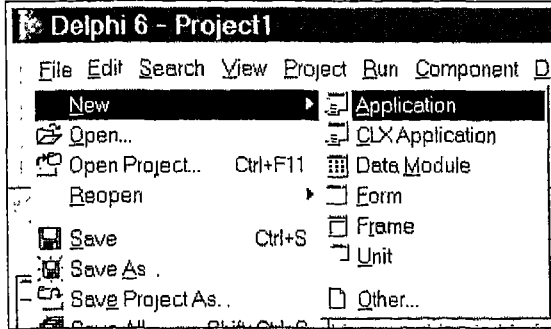
شكل توضيحي :



الأمر Application القائمة New القائمة File

يعمل هذا الأمر على إنشاء مشروع جديد يكون محتويا على فورمة فارغة ووحدة وملف مشروع.

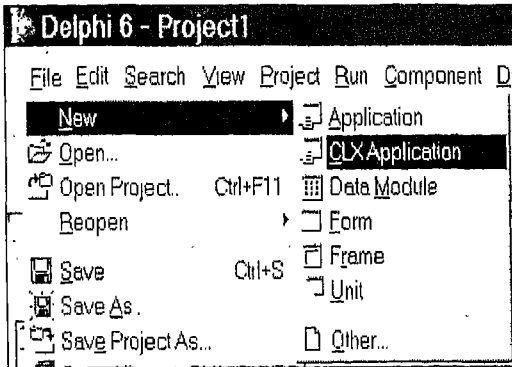
الشكل التالي يوضح لنا كيفية الوصول لهذا الأمر :



شكل توضيحي :

الأمر CLX Application القائمة New القائمة File

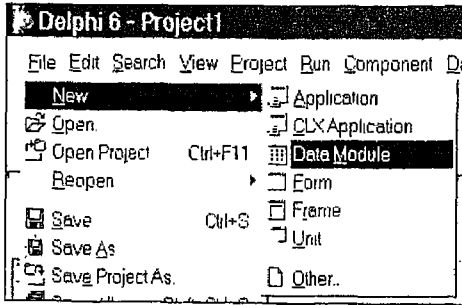
يعمل هذا الأمر على إنشاء مشروع جديد محتويا على فورمة فارغة ووحدة وملف مشروع. وهذا المشروع يكون مخصص للعمل بنظام التشغيل Linux. الشكل التالي يوضح لنا كيفية الوصول لهذا الأمر :



شكل توضيحي :

الأمر Data Module القائمة New القائمة File

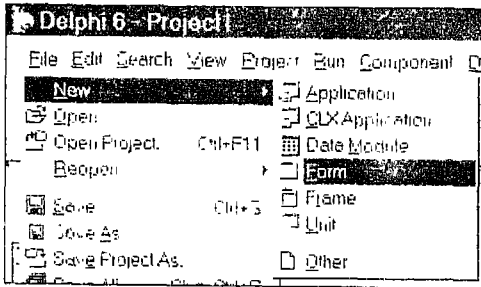
من خلال هذا الأمر يتم إنشاء Module جديد للبيانات. الشكل التالي يوضح لنا كيفية الوصول لهذا الأمر :



شكل توضيحي :

الأمر Form بالقائمة New بالقائمة File

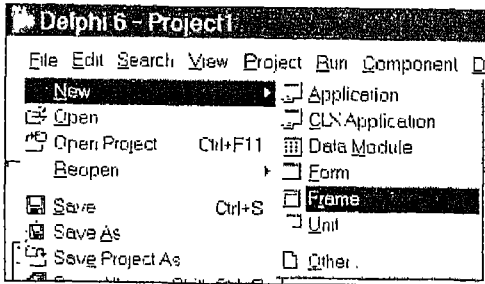
يعمل هذا الأمر على إنشاء فورمة فارغة كما يعمل على إضافتها إلى المشروع الحالي.
الشكل التالي يوضح لنا كيفية الوصول لهذا الأمر :



شكل توضيحي :

الأمر Frame بالقائمة New بالقائمة File

يعمل هذا الأمر على إنشاء إطار فارغ وإضافته إلى المشروع الحالي.
الشكل التالي يوضح لنا كيفية الوصول لهذا الأمر :

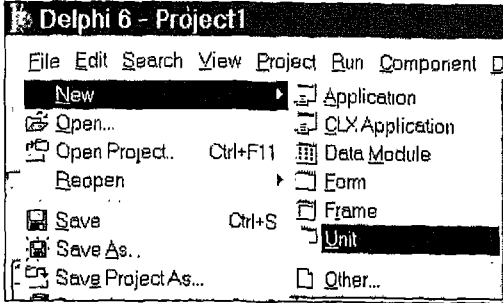


شكل توضيحي :

الأمر Unit بالقائمة New بالقائمة File

يعمل هذا الأمر على إنشاء وحدة جديدة بالمشروع الحالي.

الشكل التالى يوضح لنا كيفية الوصول لهذا الأمر :

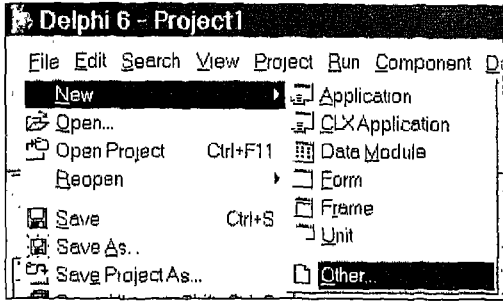


شكل توضيحي :

الأمر Other بالقائمة New بالقائمة File

يعمل هذا الأمر على فتح صندوق الحوار New Items.

الشكل التالى يوضح لنا كيفية الوصول لهذا الأمر :



شكل توضيحي :

صندوق الحوار New Items

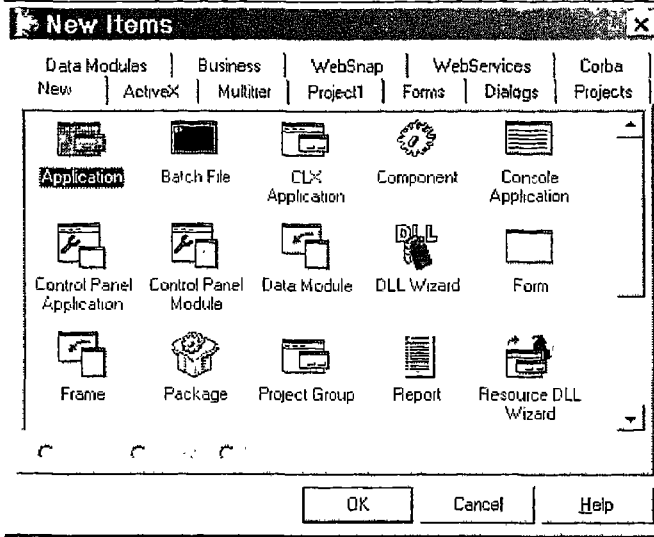
يعمل صندوق الحوار New Items على توفير وسيلة للوصول للعديد من الصفحات الموجودة بمخزن الكائنات Object Repository الذى يشتمل على صفحات مثل الصفحة Forms والصفحة Dialogs والصفحة Business التى تشتمل على تصنيفات وأقسام لعدد كبير ومتنوع من القوالب والكائنات والمعالجات التى يمكن استخدامها فى التطبيقات التى تتولى إعدادها من خلال لغة 6 Delphi.

الصفحة New بصندوق الحوار New Items

الصفحة New بصندوق الحوار New Items تشتمل على العديد من العناصر المعدة مسبقا والتى يمكنك استخدامها فى أثناء إعداد التطبيق من خلال لغة 6 Delphi.



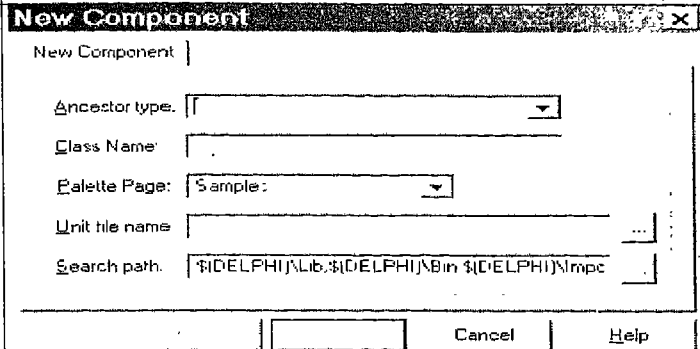
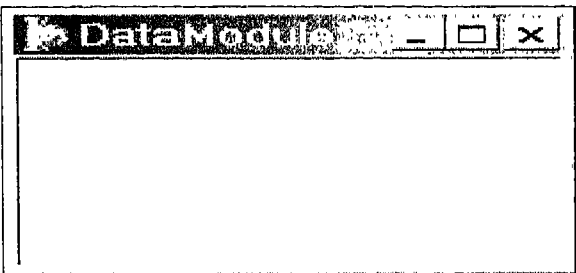
الشكل التالى يوضح لنا محتويات هذه الصفحة :

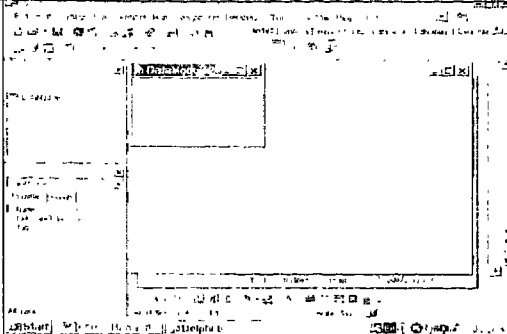
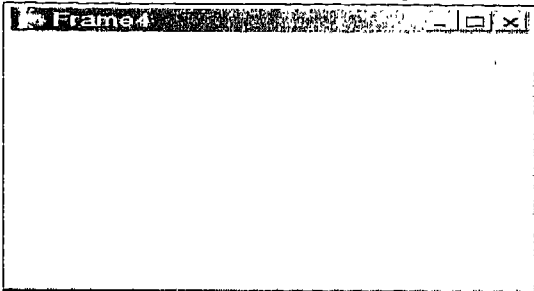
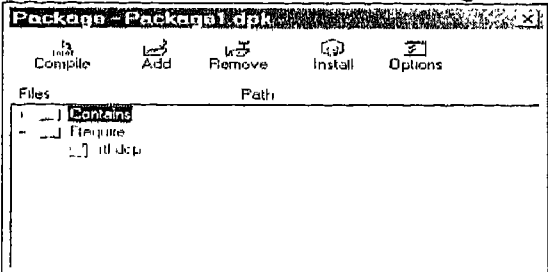


شكل توضيحي :

الجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

العنصر	الوصف والاستخدام
Application	يقوم بإنشاء مشروع جديد يحتوى على فورمة ووحدة وملف له الامتداد DPR. كما إنه يوفر لك وسيلة لاختيار قالب template.
Batch File	يقوم بإنشاء مشروع ملف حزمة جديد Batch File يكون له الامتداد BAT. وهو يسمح لك بأن تحدد حزمة من الأوامر. ومثل هذه النوعية من المشاريع لا تكون مشتملة على أى فورم كما إنها لا تشتمل على محرر الكود البرمجى Code Editor.
CLX Application	يقوم بإنشاء مشروع لدية القدرة على العمل بأنظمة التشغيل المختلفة مثل Linux ومثل هذه النوعية من المشاريع تكون مشتملة على فورمة ووحدة DPR.
Component	يقوم بإنشاء مكون جديد باستخدام المعالج Component Wizard الموضح فى الشكل التالى :

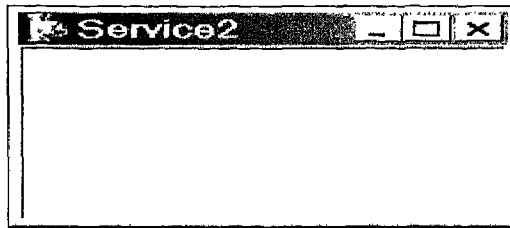
	
<p>يقوم بإنشاء مشروع تطبيق جديد يعمل من خلال نظام التشغيل DOS ولا يحتاج لأن يعمل فى بيئة الويندوز.</p>	<p>Console Application</p>
<p>يقوم بإنشاء applet جديد يمكن وضعه فى لوحة التحكم Control Panel الخاصة ببيئة الويندوز.</p>	<p>Control Panel Application</p>
<p>يقوم بإنشاء Module جديد لأى مشروع من المشاريع المصممة للعمل بلوحة التحكم ببيئة الويندوز.</p>	<p>Control Panel Module</p>
<p>يقوم بفتح Module بيانات جديد فى أداة التصميم Data Module Designer الموضحة فى الشكل التالى :</p>  <p>كما يقوم أيضا بعرض ملف الوحدة الخاص بالـ Module الجديد وذلك فى محرر الكود البرمجى Code Editor كما يعمل على إضافة هذا الـ Module إلى المشروع الحالى كما هو موضح بالشكل التالى :</p>	<p>Data Module</p>

	
<p>يقوم بإنشاء مشروع DLL (اختصار للمصطلح Dynamic Link Library) والذي نحصل منه في النهاية على ملف له امتداد DLL.</p>	<p>DLL</p>
<p>يقوم بإنشاء وإضافة فورمة فارغة إلى المشروع الحالي كما يسمح لك أيضا بأن تختار قالب فورمة.</p>	<p>Form</p>
<p>يقوم بإنشاء إطار جديد كالموضح في الشكل التالي :</p> 	<p>Frame</p>
<p>يقوم بإنشاء حزمة برمجية Package. وفي هذا الصدد نقول إن الحزمة البرمجية الجديدة تظهر في محرر الحزمة البرمجية Package Editor الموضح في الشكل التالي :</p> 	<p>Package</p>

<p>يقوم بإنشاء مجموعة مشروع Project Group جديدة لكى تشتمل على المشاريع المرتبطة معا. هذا وعن طريق إضافة المشاريع المرتبطة معا إلى مجموعة مشروع يمكنك من بناء كافة المشاريع من خلال أمر واحد. ونود هنا القول بأن ملف مجموعة المشروع يكون له الامتداد BPG.</p>	<p>Project Group</p>
<p>يقوم بإنشاء تقرير سريع يساعدك فى إنشاء تقارير قوية وفعالة لتطبيقات قواعد البيانات من خلال إمكانية التصميم على الشاشة بالنافذة Quick Report الموضحة فى الشكل التالى :</p> 	<p>Report</p>
<p>يعمل على البدء فى تشغيل معالج Wizard لكى يساعدك فى تشغيل ملفات مكتبات الربط الديناميكية DLLs التى تشتمل على إصدارات محلية من فورم التطبيقات التى تعدها. هذا والشكل التالى يوضح لنا هذا المعالج :</p> 	<p>Resource DLL Wizard</p>

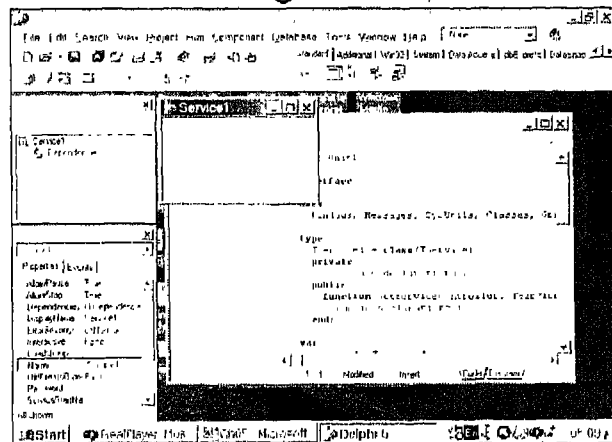


يعمل على إضافة خدمة جديدة إلى تطبيق من تطبيقات خدمات ال NT التي سبق إنشاؤها. وهذا العنصر لا يعمل على إضافة خدمات إلى تطبيق لا يكون من تطبيقات الخدمات في حين أن يمكن إضافة الكائن TService إلا إن التطبيق لن يقوم بتكوين الأحداث الخاصة بهذا الكائن كما لن يعمل الاستدعاءات المناسبة لبيئة الويندوز الخاصة بالخدمة. ونحن ننصحك هنا بأن لا تضيف أكثر من خدمة واحدة إلى أي تطبيق يكون في حاجة إلى اسم المستخدم وكلمة السر لكي يبدأ في العمل. هذا وعند النقر بالماوس نقرا مزدوجا على هذا العنصر تظهر النافذة الموضحة في الشكل التالي :



Service

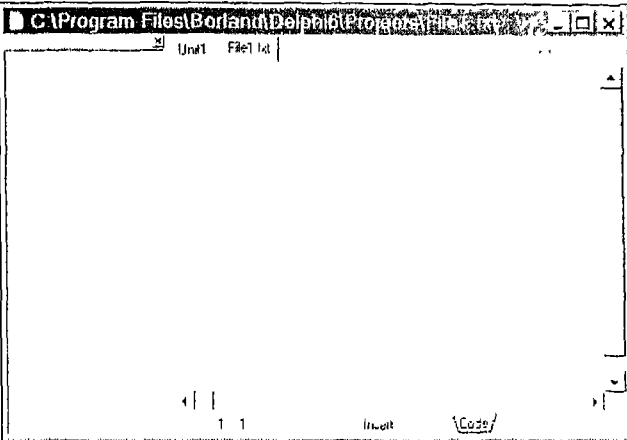
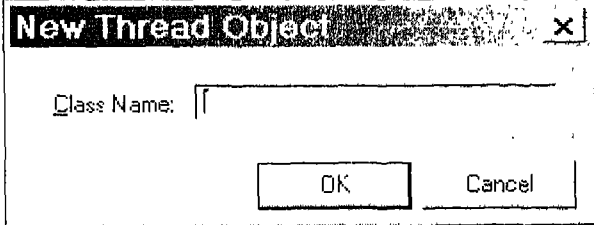
هذا العنصر يعمل على إنشاء تطبيق خدمة NT Service Application جديد. فبمجرد أن تقوم بإنشاء تطبيق خدمة فسوف تشاهد نافذة في أداة التصميم كما هو موضح بالشكل التالي :



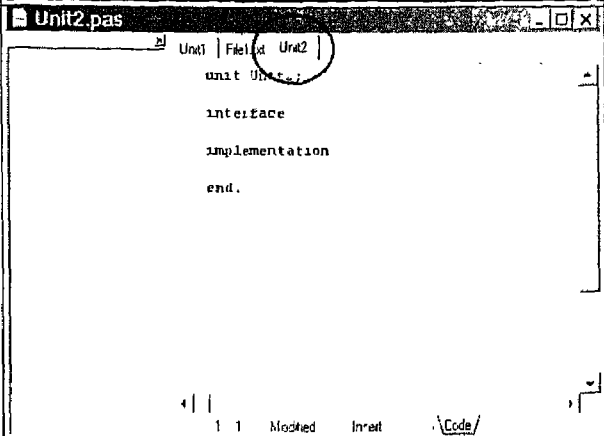
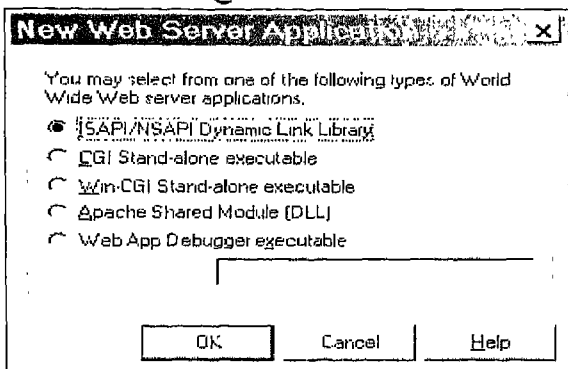
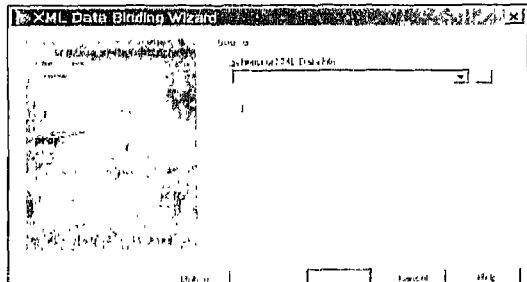
Service Application

وهذه النافذة تناظر خدمة (الكائن TService). هذا ويتم التعامل مع



<p>هذه الخدمة عن طريق تحديد قيم الخصائص الخاصة بها وذلك من خلال النافذة Object Inspector بالإضافة لإعداد الأكواد البرمجية لأدوات معاملة الأحداث المرتبطة به.</p>	
<p>يعمل هذا العنصر على إنشاء ملف نصي جديد من النوع ASCII وذلك من خلال المحرر الموضح فى الشكل التالى :</p> 	<p>Text</p>
<p>يعمل هذا العنصر على إنشاء كائن Thread جديد. وعند التعامل معه يظهر على الشاشة صندوق الحوار الموضح فى الشكل التالى :</p>  <p>والذى يسأل عن اسم القطاع الذى سينتمى إليه الكائن Thread الجديد.</p>	<p>Thread Object</p>
<p>يعمل هذا العنصر على إنشاء وإضافة وحدة جديدة إلى المشروع الحالى. وهذه الوحدة الجديدة تظهر كتنويب جديد فى محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالى :</p>	<p>Unit</p>



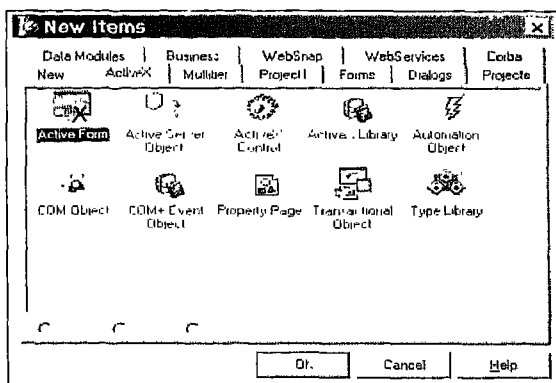
	
<p>يعمل هذا العنصر على إنشاء تطبيق Web Server Application جديد لكى يعمل فى خوادم شبكة الويب. وعند التعامل مع هذا العنصر يظهر على الشاشة صندوق الحوار الموضح فى الشكل التالى :</p> 	<p>Web Server Application</p>
<p>يعمل هذا العنصر على فتح معالج ربط بيانات الـ XML الموضح فى الشكل التالى :</p> 	<p>XML Data Binding</p>



وهذا المعالج يعتبر أداة من أجل تكوين قطاعات تعمل على تمثيل مستندات معينة معدة بلغة XML. ومثل هذه النوعية من المستندات تعمل على توفير طريقة بسيطة لتخزين المعلومات فى صورة نصية ومن ثم يكون من السهولة بمكان البحث عنها والعثور عليها.

الصفحة ActiveX بصندوق الحوار New Items

الشكل التالى يوضح لنا محتويات هذه الصفحة :



شكل توضيحى :

الكائنات الموجودة فى هذه الصفحة يتم استخدامها لإنشاء كائنات COM جديدة وفورم فعالة Active Forms وأدوات تحكم ActiveX وصفحات خصائص Property pages لأدوات التحكم ActiveX بالإضافة إلى مكتبات الأنواع الخاصة بأدوات التحكم ActiveX أو كائنات التفعيل الذاتى Automation Objects.

الجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

العنصر	الوصف والاستخدام
Active Form	يقوم هذا العنصر بإنشاء فورمة فعالة جديدة Active form والتي تعتبر أداة تحكم ActiveX بسيطة (تنحدر من TActiveForm) مهيئ مسبقا للعمل فى أى من برامج تصفح الإنترنت وشبكة الويب. هذا وعند التعامل مع هذا العنصر يظهر على الشاشة المعالج ActiveX Control Wizard الموضح فى الشكل التالى :

وهو يرشدك في أثناء عملية الإنشاء مما يسمح لك بأن تصيف أدوات تحكم إلى الفورمة. وهذا المعالج يقوم بإنشاء كل من مشروع من النوع ActiveX Library (لو كانت هناك حاجة لذلك) ومكتبة أنواع وفورمة ووحدة تنفيذ بالإضافة إلى وحدة تشتمل على إعلانات مكتبة الأنواع المناظرة. هذا وينبغي عليك ملاحظة أنه بخلاف أدوات التحكم ActiveX الأخرى فإنك لا تستطيع من خلال بيئة التطوير المتكاملة IDE التعديل في قيم خصائص فورمة Active تم بناؤها إلا إذا أضفت كود برمجى يقوم بنشر هذه الخصائص.

من خلال هذا العنصر يتم إنشاء ASP (اختصار للمصطلح Active Server Page) من أحد التطبيقات التي سبق إنشاؤها. هذا وعند التعامل مع هذا العنصر يظهر على الشاشة صندوق الحوار Active Server Object الموضح في الشكل التالي :

Active Server Object

الذى من خلاله تستطيع تحديد اسم CoClass ونموذج الـ Threading وهكذا...

من خلال هذا العنصر يتم إنشاء أداة تحكم ActiveX Control جديدة. وعند التعامل مع هذا العنصر يظهر على الشاشة المعالج ActiveX Control Wizard الموضح فى الشكل التالى :

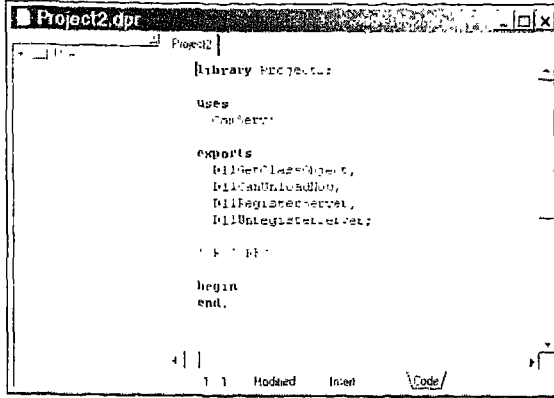
هذا المعالج يرشدك عبر عملية الإنشاء واختيار الكائن VCL الذى ترغب فى جعله الأساس لأداة التحكم الجديدة. هذا وينبغى عليك ملاحظة أن أدوات التحكم ActiveX تحتاج مكتبة ActiveX لكى يتمكنوا من امتلاك واجهات التعامل Interfaces الخاصة بهم وكذلك المعاملات التى يتم تمريرها إلى الأساليب الخاصة بهم وجعلهم يتعاملوا بفاعلية مع تطبيقات العملاء Client Applications. هذا ولو لم يتم فتح مشروع مكتبة ActiveX قبل الشروع فى إنشاء أداة تحكم ActiveX فى هذه الحالة تقوم لغة Delphi بفتح مشروع جديد من هذا النوع.

ActiveX
Control

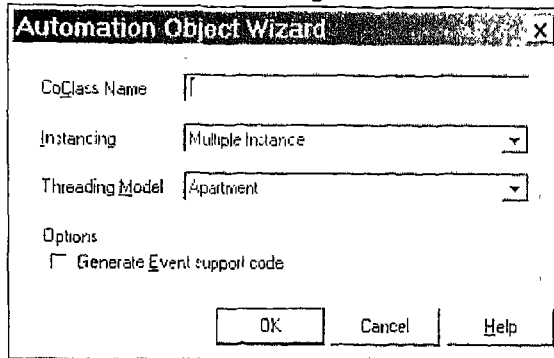
من خلال هذا العنصر يمكن إنشاء مكتبة ActiveX جديدة. وفى البداية يتم إنشاء ملف قالب يسمى Project1.dpr وهو يعد نقطة الإنطلاق بالنسبة لك للمضى قدما بعد ذلك فى عملية الإنشاء. هذا وفى حالة عدم فتح مشروع ActiveX Library قبل أن تشرع فى إنشاء أداة تحكم ActiveX فى هذه الحالة تقوم لغة Delphi بفتح مشروع جديد من هذه النوعية الذى يكون مشتملا على محرر الكود

ActiveX
Library

البرمجي Code Editor فقط كما هو موضح بالشكل التالي :

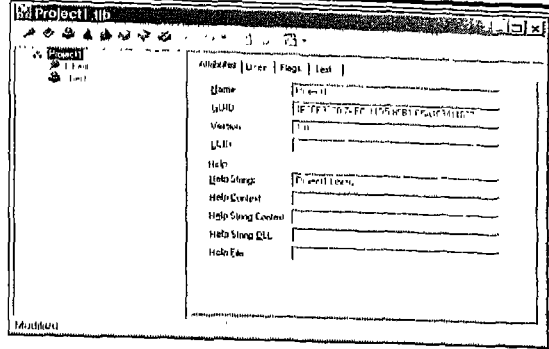
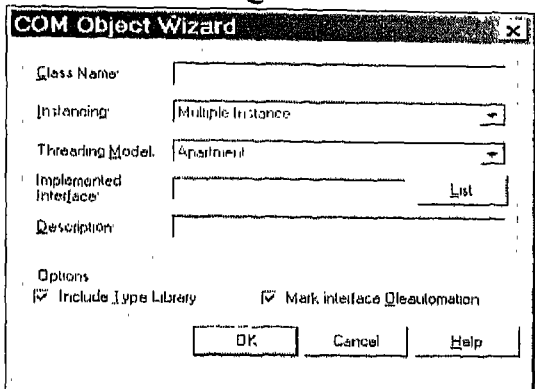


يعمل هذا العنصر على إنشاء كائن تفعيل ذاتي Automation جديد. وعند التعامل مع هذا العنصر يظهر على الشاشة المعالج Automation Object Wizard في الشكل التالي :



هذا المعالج يسمح لك بأن تدخل اسم قطاع للكائن Automation الجديد بالإضافة إلى إمكانية تحديد نوعية حالة الكائن وكذلك نموذج الـ Threading. هذا وبمجرد أن يقوم المعالج بإنشاء الكائن Automation فإنه يتم استخدام محرر مكتبة النوع Type Library الموضح في الشكل التالي :

Automation
Object

 <p>وذلك لتعريف واجهة الاستخدام التي سيستخدمها الكائن للعمل في تطبيقات العملاء.</p>	
<p>من خلال هذا العنصر يمكن إنشاء كائن COM جديد الذي يستخدم في إعداد واجهة استخدام مرخصة أو إنشاء واجهة استخدام جديدة تنحدر من IUnknown. هذا وعند التعامل مع هذا العنصر يظهر على الشاشة المعالج COM Object Wizard الموضح في الشكل التالي :</p>  <p>هذا المعالج يسمح لك بأن تحدد خصائص الخادم COM الجديد.</p>	COM Object
<p>من خلال هذا العنصر يتم إنشاء كائن COM+ Event جديد الذي يعمل على دفع أحداث الخادم لكافة العملاء المسجلة والمرخصة. وعند التعامل مع هذا العنصر يظهر على الشاشة المعالج COM+ Event Object Wizard الموضح في الشكل التالي :</p>	COM+ Event Object



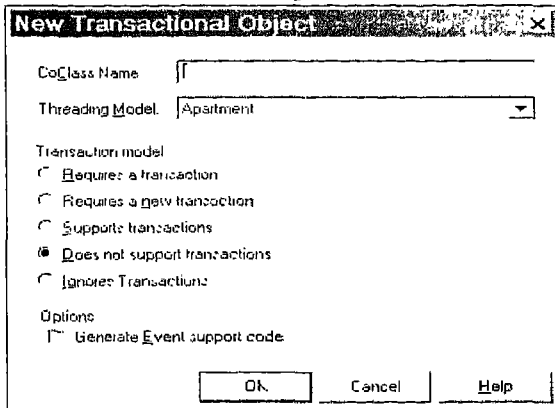
هذا المعالج يضيف كائن COM+ event إلى مشروع event object موجود بالفعل أو يبدأ في مشروع event object جديد من أجل هذا الكائن. وبعد أن يقوم المعالج بتكوين event object فإنه يستخدم محرر مكتبة النوع لتعريف واجهة الاستخدام الخاصة بالكائن COM+ event كما هو موضح بالشكل التالي :

من خلال هذا العنصر يمكن إنشاء ملف يعمل على تهيئة وإعداد صفحة الخاصية لأداة تحكم ActiveX. وصفحة الخاصية تظهر في مود التصميم وتكون جاهزة بحيث يمكنك من إضافة إعلانات خاصة وعامة. وأنت تستطيع تصميم صندوق حوار في نافذة الفورمة كما يمكن تجميع الخصائص في مجموعة واحدة وذلك لجعل من السهولة بمكان، بالنسبة للمبرمجين أن يقوموا بتعديل أداة التحكم عند التعامل معها في أي تطبيق.

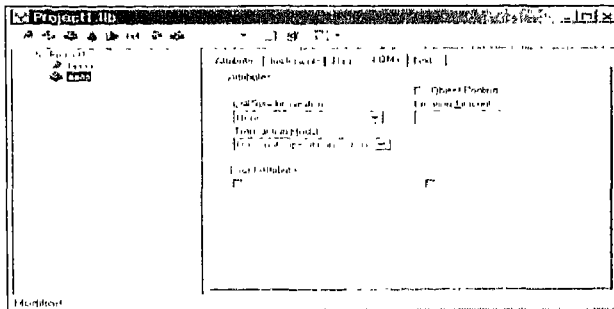
Property Page

Transactional Object

من خلال هذا العنصر يمكن إنشاء كائن تفعيل ذاتى Automation يتمتع بخدمات التطبيقات الموزعة التى يوفرها MTS أو COM+. هذا وعند التعامل مع هذا العنصر يظهر على الشاشة المعالج Transactional Object Wizard الموضح فى الشكل التالى :



هذا المعالج يسمح لك بأن تحدد نموذج الحركة المطلوب. وبعد أن ينتهى التعامل مع المعالج فإنه يتم استخدام محرر مكتبة النوع لتعريف واجهة الاستخدام التى سيستخدمها الكائن للتعامل مع تطبيقات العملاء. ونود هنا القول بأن لو كان من المفروض أن يتم تركيب هذا الكائن تحت COM+ فى هذه الحالة استخدم الصفحة COM+ بمحرر مكتبة النوع الموضحة فى الشكل التالى :



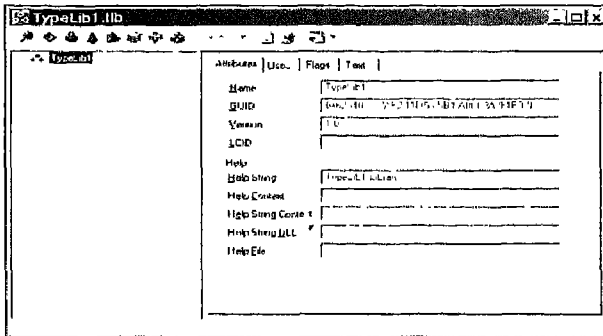
وذلك لتحديد مجموعة الاختيارات التى سيتم استخدامها عند تركيب الكائن داخل أى من التطبيقات COM+.



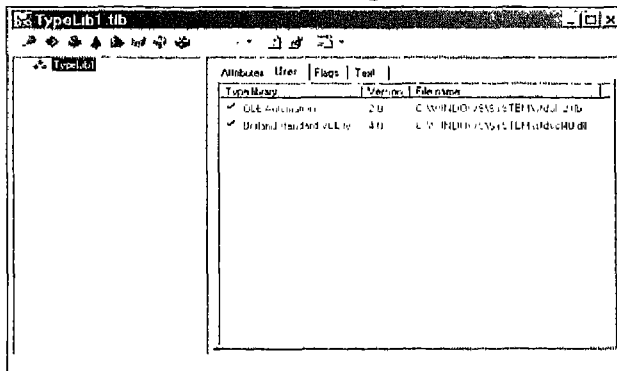
Type Library

من خلال هذا العنصر يمكن إنشاء أو تحرير أى مكتبة من مكتبات النوع التى يمكن استخدامها بواسطة مكتبات أنواع أخرى أو التى تعمل على تعريف واجهات الاستخدام التى يستخدمها المعالج COM Object Wizard لتكوين الكائنات. هذا وعند التعامل مع هذا العنصر يظهر على الشاشة مباشرة محرر مكتبة النوع Type Library editor ولكن هنا نجد أن المحرر-وليس كما سبق- يحتوى على أربعة تبويبات وهى :

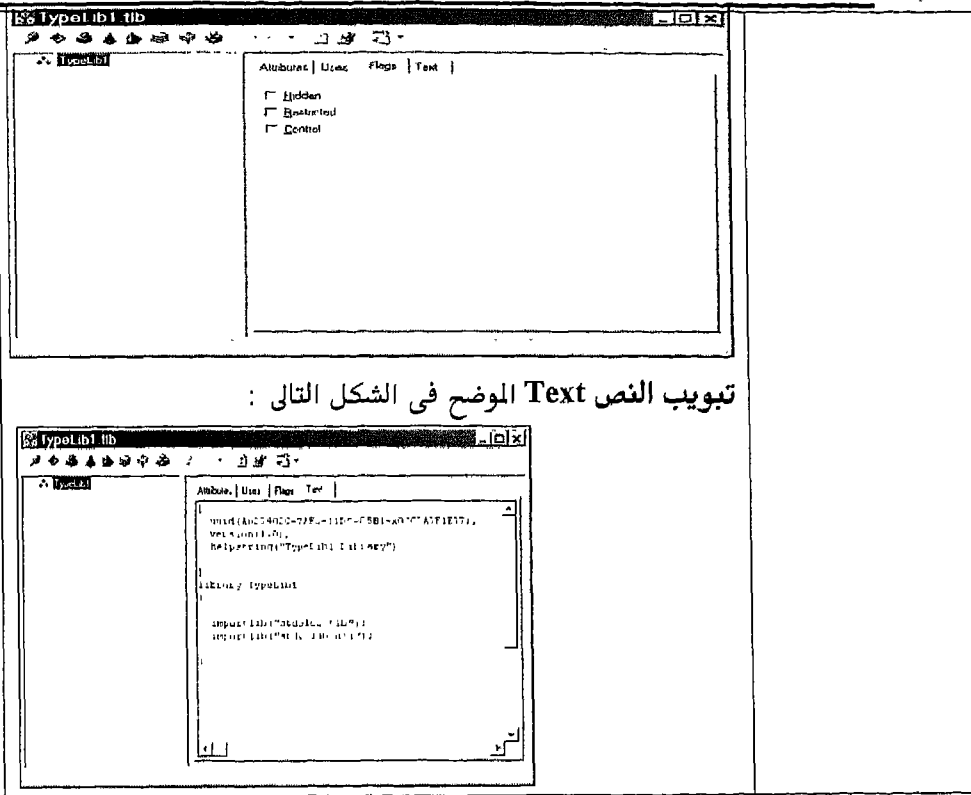
تبويب الصفات Attributes الموضح فى الشكل التالى :



تبويب المستخدمين Users الموضح فى الشكل التالى :



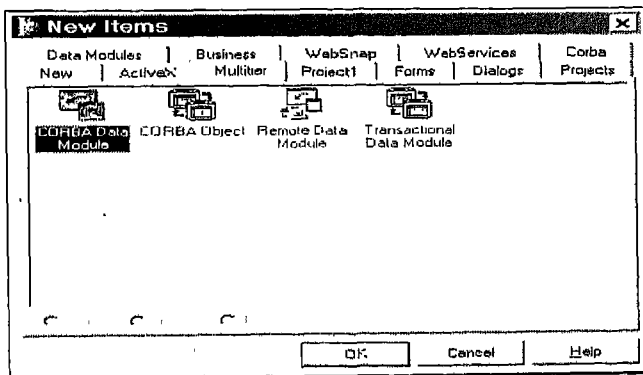
تبويب الإشارات Flags الموضح فى الشكل التالى :



تبويب النص Text الموضح في الشكل التالي :

الصفحة Multitier بصندوق الحوار New Items

الشكل التالي يوضح لنا محتويات هذه الصفحة :

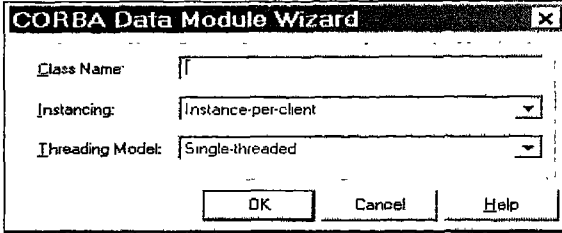
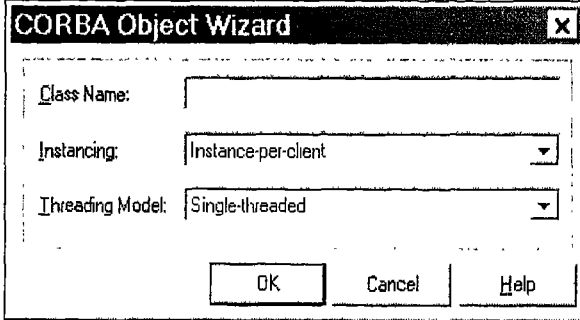


شكل توضيحي :

الكائنات والعناصر الموجودة في هذه الصفحة يمكن استخدامها لإنشاء خوادم Servers تعتبر جزء من التطبيقات المتعددة الطبقات.



الجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

العنصر	الوصف والاستخدام
CORBA Data Module	<p>من خلال هذا العنصر يتم إنشاء CORBA Data Module لكى يعمل على أساس كونه خادم فى تطبيق قواعد بيانات متعدد الطبقات multi-tiered يستخدم CORBA على أساس كونها بروتوكولات إتصال. هذا ويتم التعامل مع هذا العنصر من خلال المعالج CORBA Data Module Wizard الموضح فى الشكل التالى :</p>  <p>هذا المعالج يقوم بتكوين التنفيذ الإبتدائى وتعريف قطاع تنفيذ لـ Module البيانات مع بعض الخيارات الخاصة بكل من عملية threading وعملية instancing.</p>
CORBA Object	<p>يعمل هذا العنصر على إنشاء كائن CORBA Server. هذا ويتم التعامل مع هذا العنصر من خلال المعالج CORBA Object Wizard الموضح فى الشكل التالى :</p> 



من خلال هذا العنصر يتم إنشاء تطبيق Server فى تطبيق قاعدة بيانات متعدد الطبقات قائم على التكنولوجيا COM. هذا ويتم التعامل مع هذا العنصر من خلال المعالج Remote Data Module Wizard الموضح فى الشكل التالى :

وهذا المعالج يعمل على تكوين التنفيذ الابتدائى وتعريف قطاع التنفيذ لـ Module البيانات مع خيارات خاصة بكل من العملية Instancing والعملية Threading.

Remote Data Module

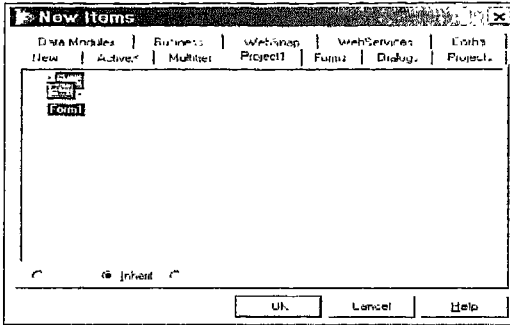
من خلال هذا العنصر يتم إنشاء Module للبيانات التى توجد فى مصادر بعيدة Remote Data Module والذى يمكن أن يستفيد من الخدمات التى يتم توفيرها من خلال MTS أو COM+. هذا ويتم التعامل مع هذا العنصر من خلال المعالج Transactional Data Module Wizard الموضح فى الشكل التالى :

Transactional Data Module

وهذا المعالج يعمل على تكوين التنفيذ الابتدائي وتعريف قطاع التنفيذ لـ Module البيانات مع خيارات خاصة بكل من العملية Threading و Instancing.

الصفحة Project بصندوق الحوار New Items

الشكل التالي يوضح لنا محتويات هذه الصفحة :

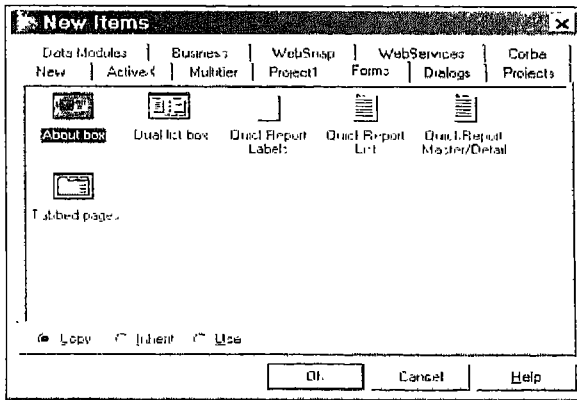


شكل توضيحي :

لو أن لديك مشروع مفتوح حالياً ستجد أن الصفحة Project في صندوق الحوار New Items قد أصبح اسمها هو نفس اسم المشروع المفتاح حالياً. كذلك تجد أيضاً أن هذه الصفحة تضم كافة الفورم الموجودة بهذا المشروع. وأنت تستطيع إنشاء فورمة وريثة من أى فورمة فى أى مشروع سبق إعداده.

الصفحة Forms بصندوق الحوار New Items

الشكل التالي يوضح لنا محتويات هذه الصفحة :

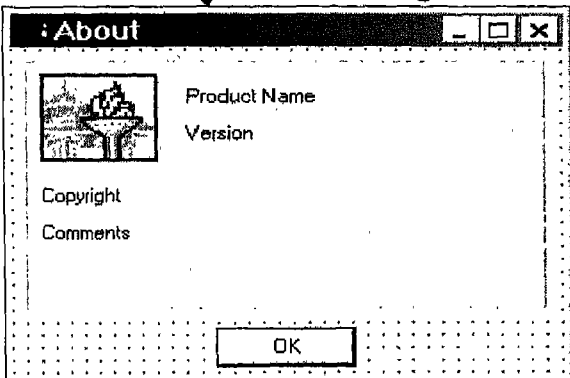
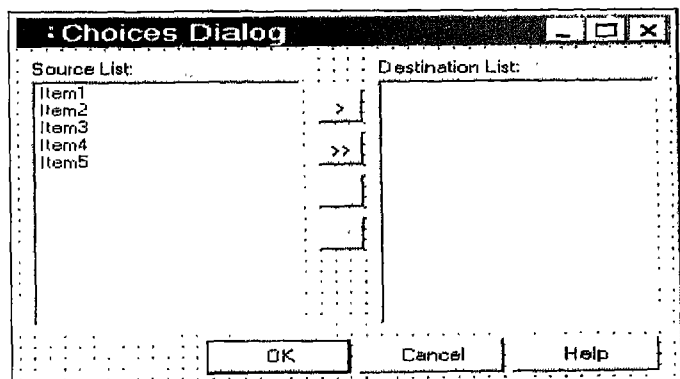


شكل توضيحي :



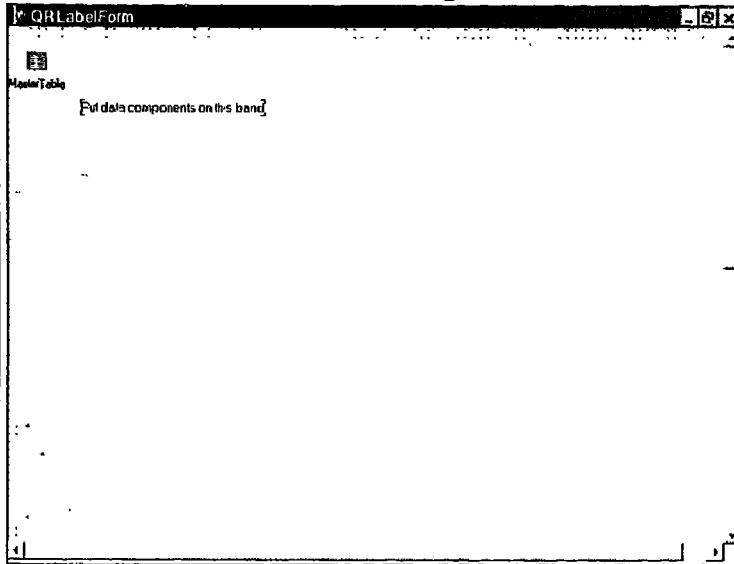
من خلال العناصر الموجودة فى الصفحة Forms يمكن إنشاء أنواع مختلفة من الفورم فى المشروع الذى يتم التعامل معه حالياً.

الجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

الوصف والاستخدام	العنصر
<p>من خلال هذا العنصر يتم إنشاء صندوق حوار About الذى يعرض معلومات عن التطبيق مثل الاسم ورقم الإصدار Version وتاريخ التصميم... والشكل التالى يوضح لنا صندوق الحوار هذا :</p> 	<p>About Box</p>
<p>من خلال هذا العنصر يتم إنشاء فورمة تشتمل على إثنيين من قوائم العرض والذى يمكن تبادل العناصر بينهما كما هو موضح بالشكل التالى :</p> 	<p>Dual List Box</p>

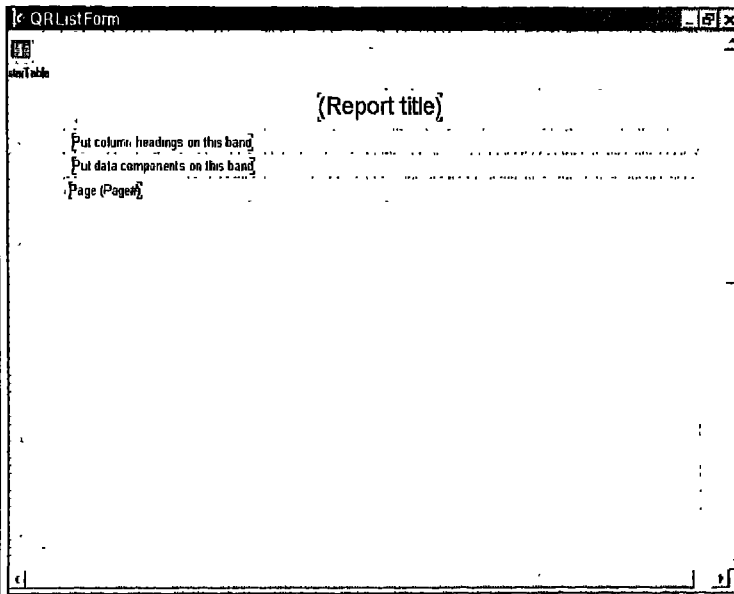


من خلال هذا العنصر يتم بطريقة سريعة إنشاء الحقول التى يتألف منها أى تقرير كما هو موضح بالشكل التالى :



QuickReport
Labels

من خلال هذا العنصر يمكن إنشاء تقرير ملخص بشكل سريع من خلال النافذة الموضحة فى الشكل التالى :

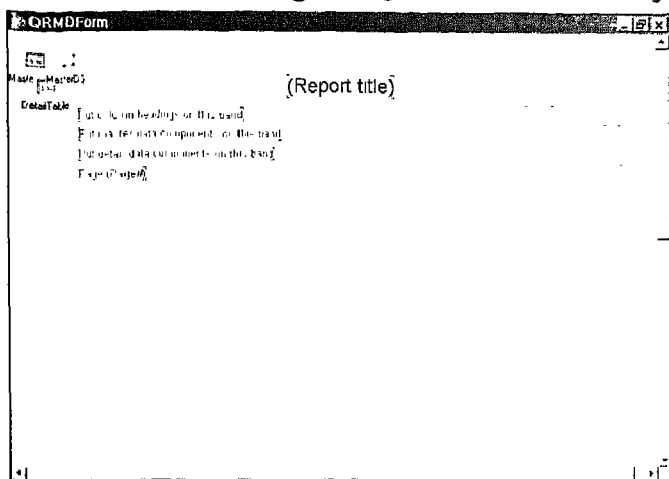


QuickReport
List



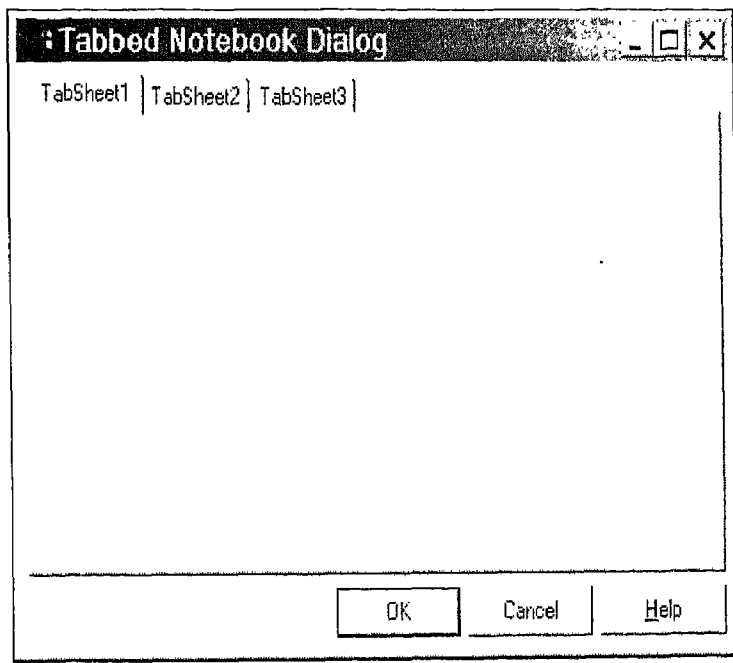
من خلال هذا العنصر يمكن إنشاء تقارير تفصيلية وأساسية بشكل سريع وذلك من خلال النافذة الموضحة في الشكل التالي :

QuickReport
Master/Detail



من خلال هذا العنصر يتم إنشاء فورمة تتألف من عدة تبويبات أو صفحات كالفورمة الموضحة في الشكل التالي :

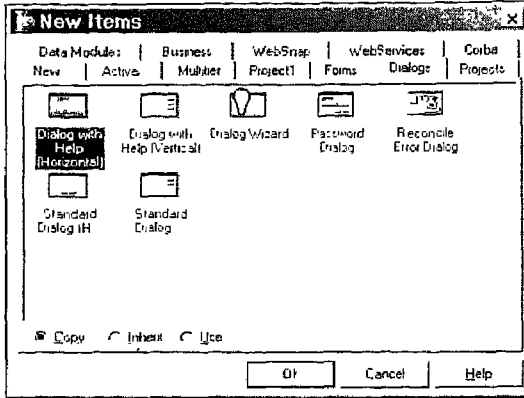
Tabbed Pages





الصفحة Dialogs بصندوق الحوار New Items

الشكل التالى يوضح لنا محتويات هذه الصفحة :



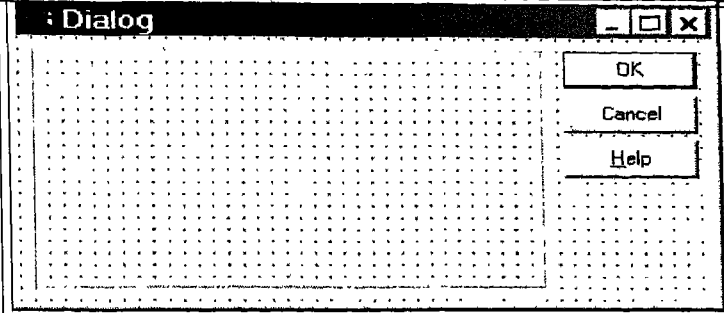
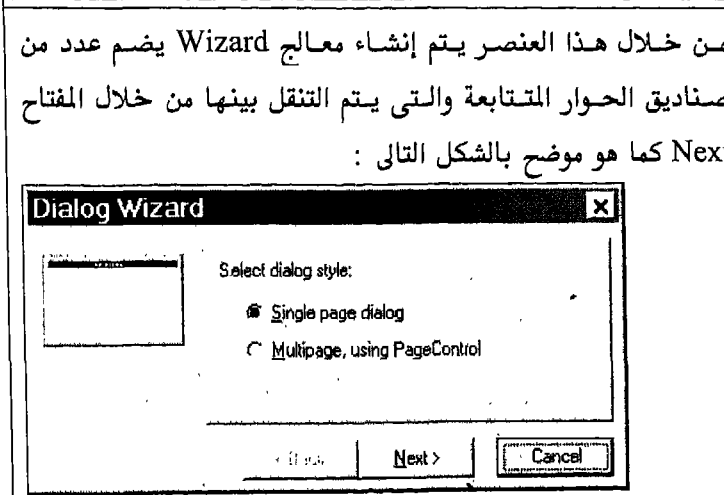
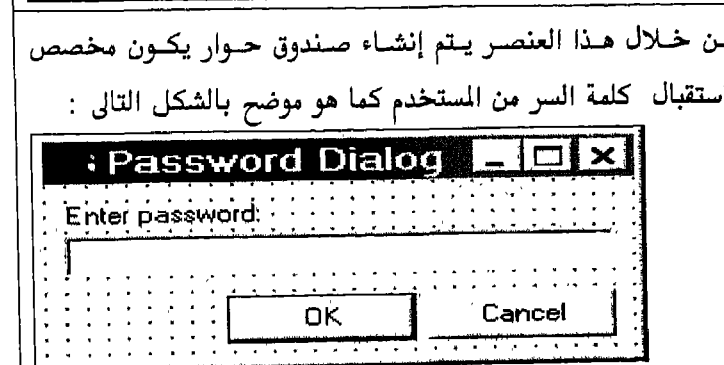
شكل توضيحي :

من خلال العناصر الموجودة فى هذه الصفحة يمكن إنشاء أنواع كثيرة من صناديق الحوار.

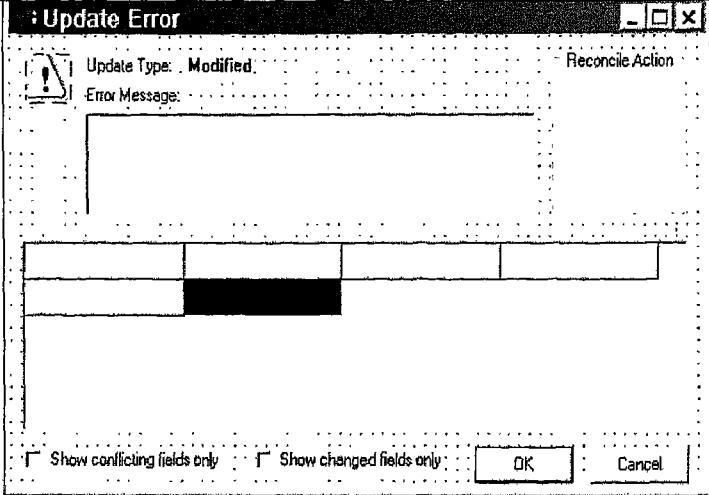
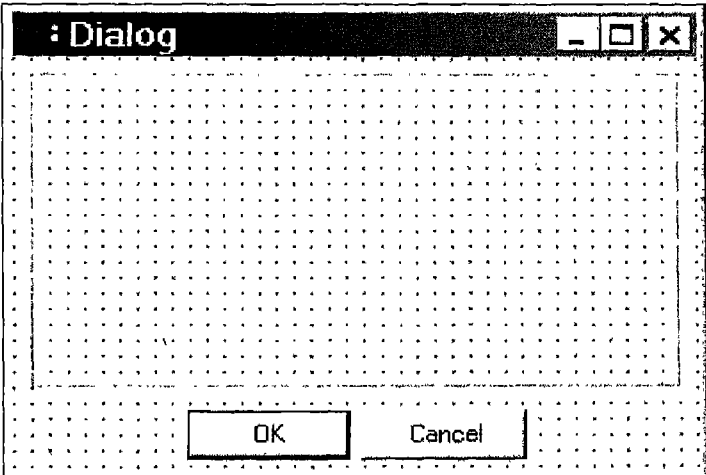
الجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

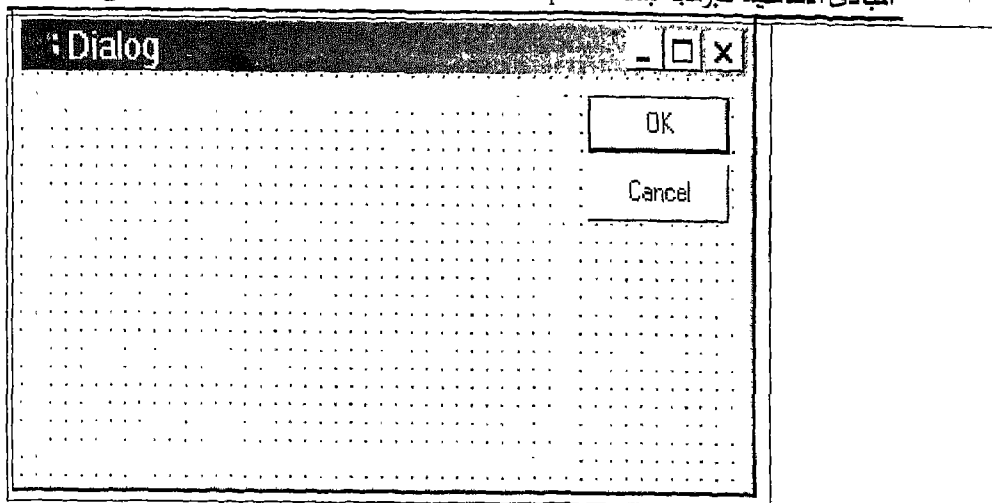
العنصر	الوصف والاستخدام
Dialog With Help (Horizontal)	<p>من خلال هذا العنصر يتم إنشاء صندوق حوار يكون مشتمل على مجموعة من المفاتيح بجوار بعضها البعض فى الاتجاه الأفقى كما هو موضح بالشكل التالى :</p> 
Dialog With Help (Vertical)	<p>من خلال هذا العنصر يتم إنشاء صندوق حوار يكون مشتمل على مجموعة من المفاتيح فوق بعضها البعض فى الاتجاه الرأسى كما هو موضح بالشكل التالى :</p>



	
<p>من خلال هذا العنصر يتم إنشاء معالج Wizard يضم عدد من صناديق الحوار المتتابعة والتي يتم التنقل بينها من خلال المفتاح Next كما هو موضح بالشكل التالي :</p> 	<p>Dialog Wizard</p>
<p>من خلال هذا العنصر يتم إنشاء صندوق حوار يكون مخصص لاستقبال كلمة السر من المستخدم كما هو موضح بالشكل التالي :</p> 	<p>Password Dialog</p>
<p>من خلال هذا العنصر يتم إنشاء صندوق حوار يتم من خلاله عرض مجموعة من رسائل الخطأ التي يتم تحديثها كما هو موضح بالشكل التالي :</p>	<p>Reconcile Error Dialog</p>

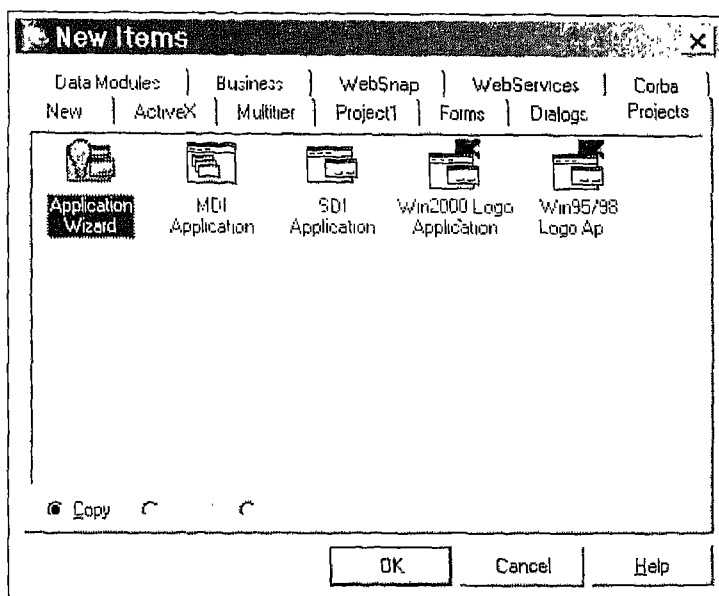


	
<p>من خلال هذا العنصر يتم إنشاء صندوق حوار قياسي لا يشتمل على المفتاح Help كما المفاتيح التي به تكون مرصوفة أفقيا كما هو موضح بالشكل التالي :</p> 	<p>Standard Dialog (Horizontal)</p>
<p>من خلال هذا العنصر يتم إنشاء صندوق حوار قياسي لا يشتمل على المفتاح Help كما المفاتيح التي به تكون مرصوفة رأسيا كما هو موضح بالشكل التالي :</p>	<p>Standard Dialog (Vertical)</p>



الصفحة Projects بصندوق الحوار New Items

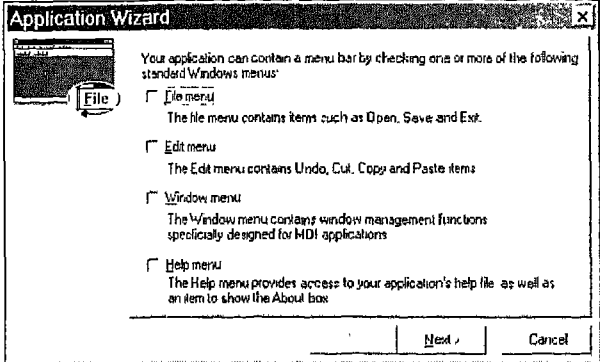
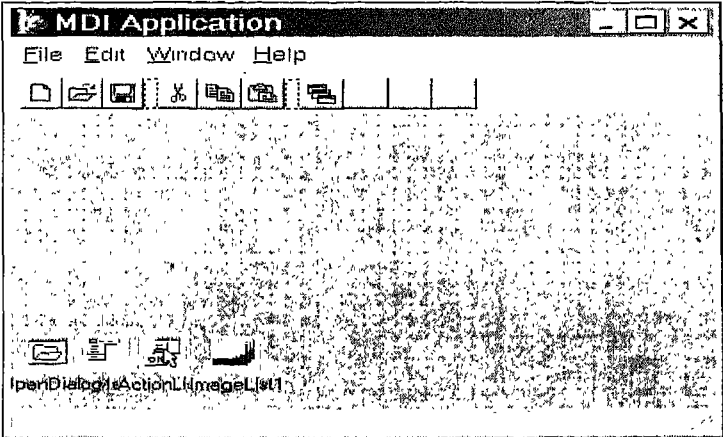
الشكل التالي يوضح لنا محتويات هذه الصفحة :



شكل توضيحي :

من خلال العناصر الموجودة بهذه الصفحة يمكن إنشاء أنواع عديدة من المشاريع والتطبيقات. هذا والجدول التالي يقدم لنا وصف مختصر للعناصر الموجودة في هذه الصفحة :

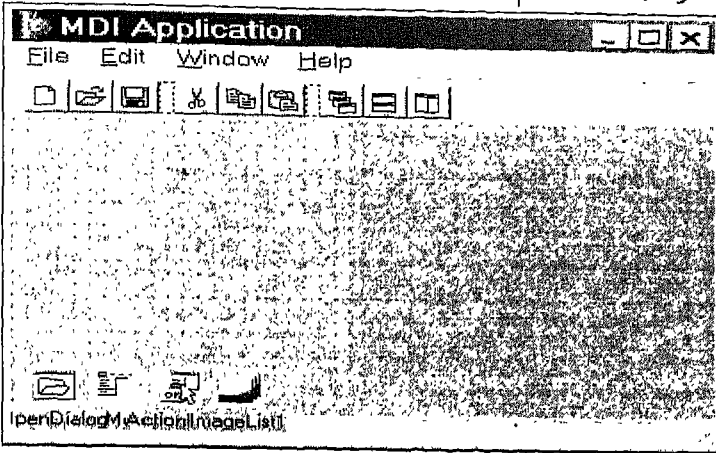


الوصف والاستخدام	العنصر
<p>من خلال هذا العنصر يتم إنشاء تطبيق بطريقة سهلة وبسيطة وذلك عن طريق التعامل مع معالج التطبيق Application Wizard الموضح فى الشكل التالى :</p> 	Application Wizard
<p>من خلال هذا العنصر يتم إنشاء تطبيق من النوع MDI (اختصار للمصطلح Multiple Document Interface) مثل العديد من التطبيقات المشهورة مثل برنامج Word. والتطبيق الذى نحصل عليه يمكن تشغيله من خلال بيئة الويندوز والعديد من أنظمة التشغيل الأخرى مثل Linux. هذا والشكل التالى يوضح لنا واجهة الاستخدام الخاصة بهذه النوعية من التطبيقات :</p> 	CLX MDI Application



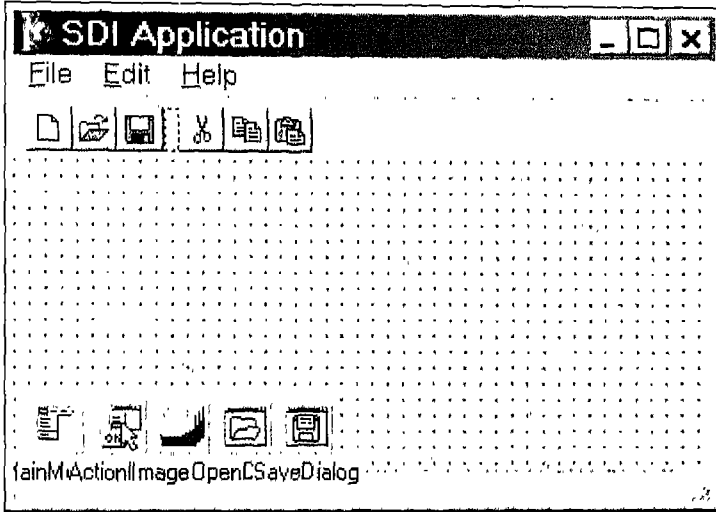
من خلال هذا العنصر يتم إنشاء تطبيق من النوع MDI (اختصار للمصطلح Multiple Document Interface) مثل العديد من التطبيقات المشهورة مثل برنامج Word. هذا والشكل التالى يوضح لنا واجهة الاستخدام الخاصة بهذه النوعية من التطبيقات :

MDI Application



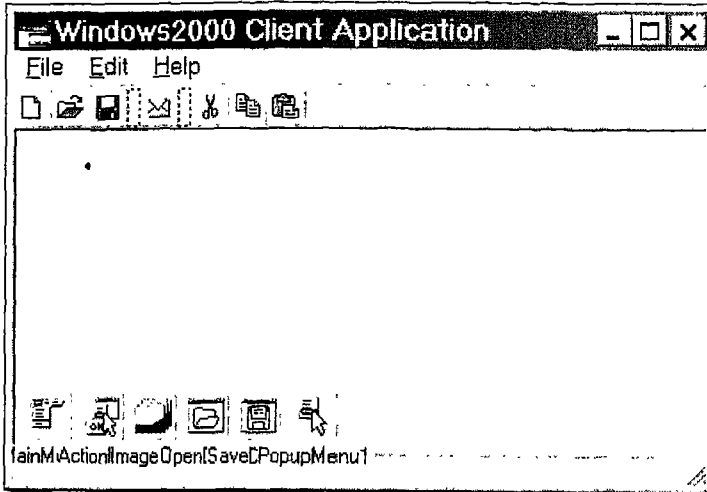
من خلال هذا العنصر يتم إنشاء تطبيق من النوع SDI (اختصار للمصطلح Single Document Interface) مثل برنامج WordPad الذى يأتى مع بيئة الويندوز. هذا والشكل التالى يوضح لنا واجهة الاستخدام الخاصة بهذه النوعية من التطبيقات :

SDI Application



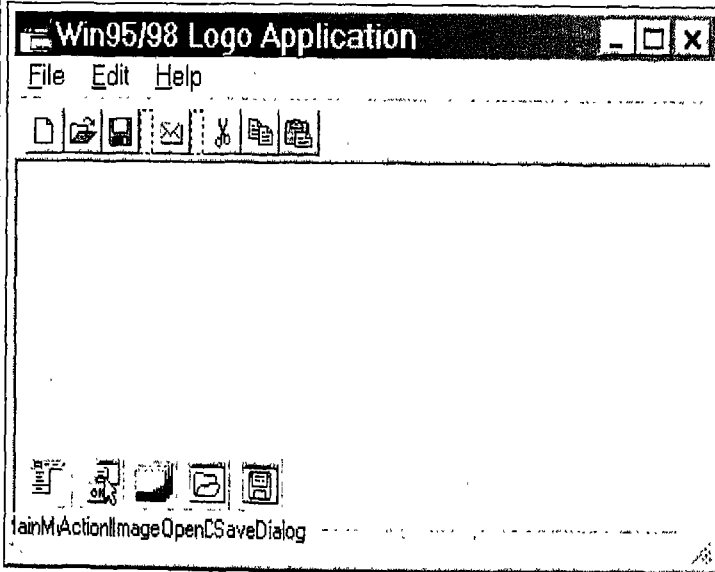


من خلال هذا العنصر يتم إنشاء تطبيقات تحمل بعض صفات بيئة ويندوز ٢٠٠٠ ومثل هذه التطبيقات يكون لها واجهة استخدام كالموضحة في الشكل التالي :



Win 2000
Logo
Application

من خلال هذا العنصر يتم إنشاء تطبيقات تحمل بعض صفات بيئة ويندوز ٩٥ أو ٩٨ ومثل هذه التطبيقات يكون لها واجهة استخدام كالموضحة في الشكل التالي :

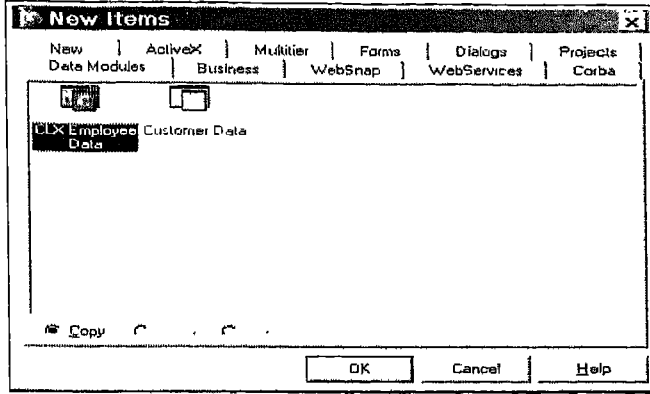


Win 95/98
Logo
Application



الصفحة Data Modules بصندوق الحوار New Items

الشكل التالي يوضح لنا محتويات هذه الصفحة :

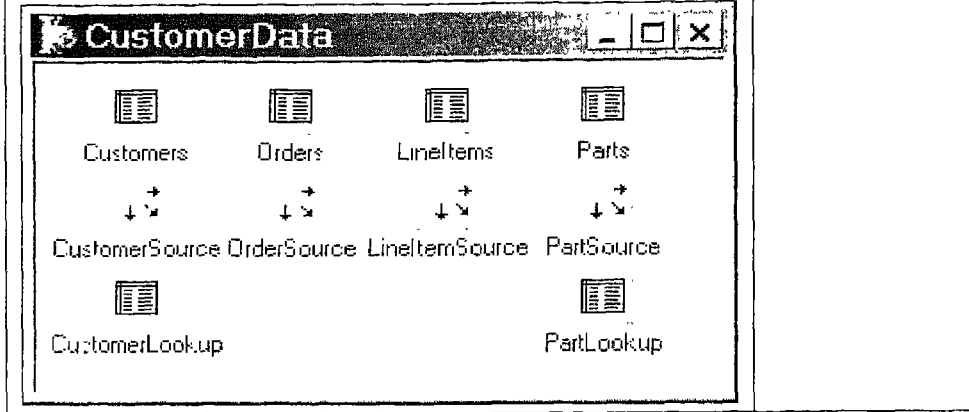


شكل توضيحي :

من خلال العناصر الموجودة في هذه الصفحة نستطيع إنشاء نوعين مختلفين من Modules البيانات. هذا والجدول التالي يقدم لنا وصف مختصر للعناصر الموجودة في هذه الصفحة :

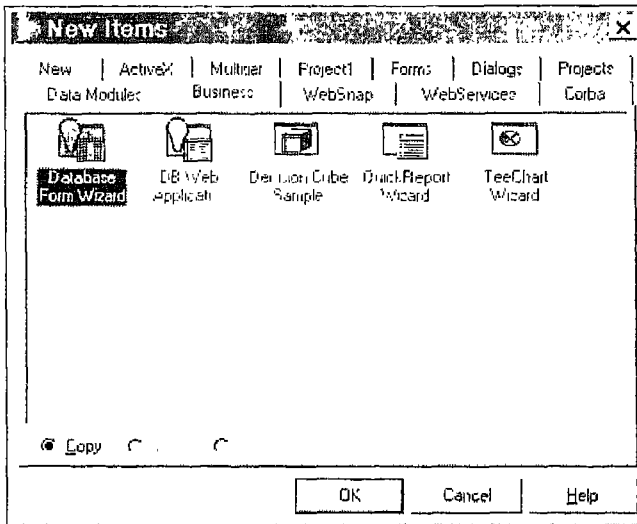
العنصر	الوصف والاستخدام
CLX Employee Data	<p>من خلال هذا العنصر يتم إنشاء Module لتوظيف البيانات في التطبيقات التي من النوع CLX وهي التطبيقات التي تكون مهيئة للعمل بكل من بيئة الويندوز وأنظمة التشغيل الأخرى مثل Linux. هذا ويتم التعامل مع هذا العنصر من خلال النافذة الموضحة في الشكل التالي :</p>

من خلال هذا العنصر يتم تفصيل Module للبيانات وذلك من خلال النافذة الموضحة في الشكل التالي :



الصفحة Business بصندوق الحوار New Items

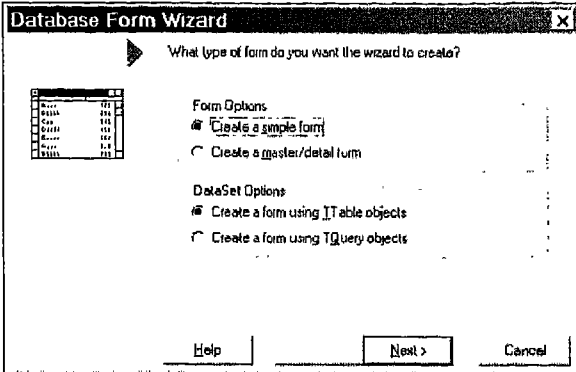
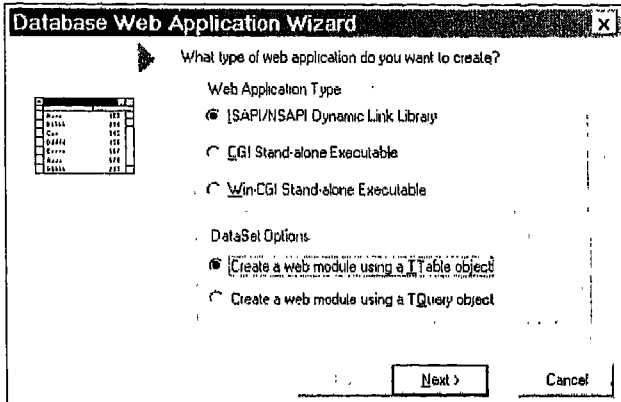
الشكل التالي يوضح لنا محتويات هذه الصفحة :



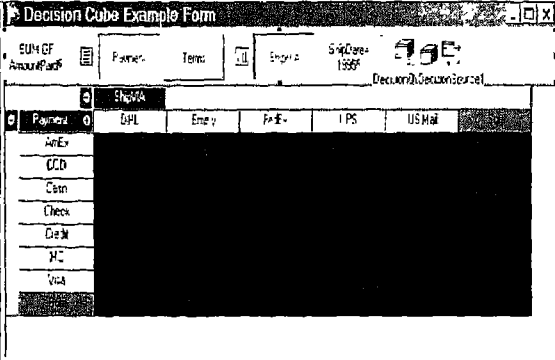
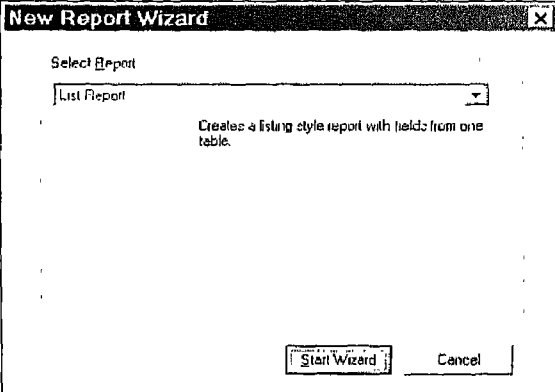
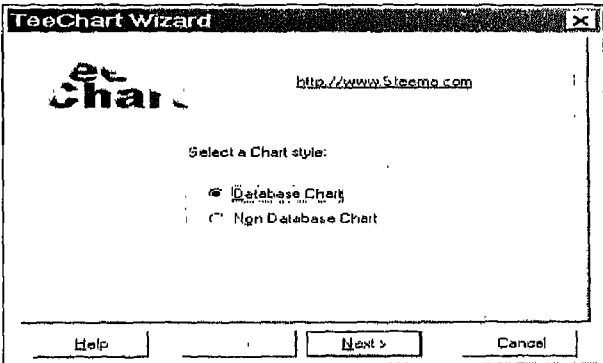
شكل توضيحي :

من خلال العناصر الموجودة في هذه الصفحة يمكن إنشاء أنواع مختلفة من التطبيقات التجارية مثل تطبيقات قواعد البيانات وتطبيقات الويب وتطبيقات دعم اتخاذ القرار وتطبيقات الإعداد السريع لتقارير العمل بالإضافة إلى تطبيقات التمثيل البياني. هذا والجدول التالي يقدم لنا وصف مختصر للعناصر الموجودة في هذه الصفحة :



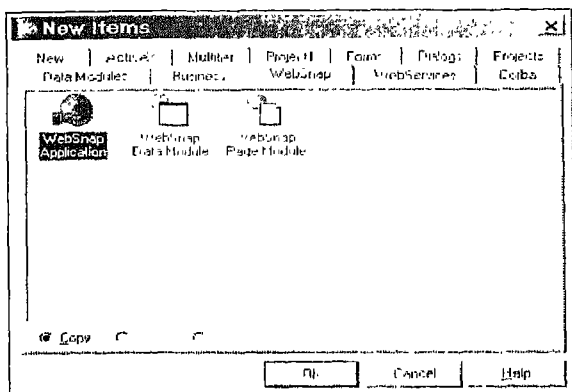
الوصف والاستخدام	العنصر
<p>من خلال العنصر يتم إنشاء فورمة لتطبيق قواعد بيانات وذلك من خلال المعالج Database Form Wizard الذى يظهر على الشاشة والموضح بالشكل التالى :</p> 	<p>Database Form Wizard</p>
<p>من خلال العنصر يتم إنشاء تطبيق قواعد بيانات يعمل بشبكة الويب وذلك من خلال المعالج DB Web Application Wizard الموضح فى الشكل التالى :</p> 	<p>DB Web Application Wizard</p>
<p>من خلال العنصر يتم إنشاء نموذج مبسط للتطبيقات التى تعمل على دعم إتخاذ القرار. هذا والشكل التالى يوضح لنا النموذج الذى نحصل عليه من هذا العنصر :</p>	<p>Decision Cube Sample</p>



		
<p>من خلال العنصر يتم إنشاء تقارير عمل سريعة وذلك من خلال المعالج QuickReport Wizard الموضح في الشكل التالي :</p> 		QuickReport Wizard
<p>من خلال العنصر يتم إنشاء أشكال ورسومات بيانية وذلك من خلال المعالج TeeChart Wizard الموضح في الشكل التالي :</p> 		TeeChart Wizard

الصفحة WebSnap بصندوق الحوار New Items

الشكل التالي يوضح لنا محتويات هذه الصفحة :



شكل توضيحي :

من خلال العناصر الموجودة في هذه الصفحة يمكن إنشاء أنواع مختلفة من تطبيقات الت، تعمل في شبكة الويب. هذا والجدول التالي يقدم لنا وصف مختصر للعناصر الموجودة في هذه الصفحة :

العنصر	الوصف والاستخدام
WebSnap Application	<p>عن طريق هذا العنصر يمكن إنشاء تطبيقات تعمل في شبكة الويب وذلك من خلال صندوق الحوار New WebSnap Application الموضح في الشكل التالي :</p>

عن طريق هذا العنصر يمكن إنشاء Module للبيانات يتم استخدامه مع التطبيقات التى تعمل بشبكة الويب. وهذا الإنشاء يتم من خلال صندوق الحوار New WebSnap Data Module الشكل التالى :

WebSnap Data Module

New WebSnap Data Module

Module Options

Creation:

Caching:

عن طريق هذا العنصر يمكن إنشاء WebSnap Page Module لاستخدامها مع التطبيقات التى تعمل بشبكة الويب. وهذا الإنشاء يتم من خلال صندوق الحوار New WebSnap Page Module الموضح فى الشكل التالى :

WebSnap Page Module

New WebSnap Page Module

Producer Type:

Script Engine:

HTML New File: ☒

Template:

Page Name:

Title:

Published: ☒ Login Required: ☐

Module Options

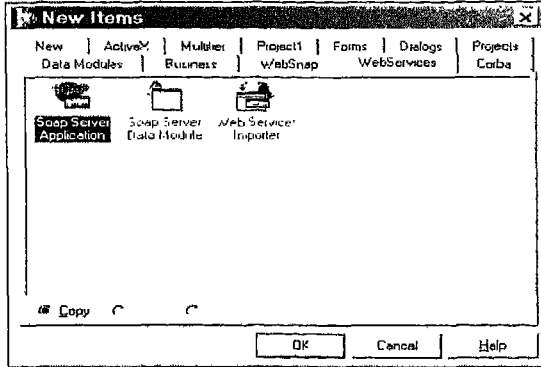
Creation:

Caching:

☐ Default

الصفحة Web Services بصندوق الحوار New Items

الشكل التالي يوضح لنا محتويات هذه الصفحة :



شكل توضيحي :

من خلال العناصر الموجودة في هذه الصفحة يمكن إنشاء تطبيقات وخدمات تعمل بخوادم شبكة الويب . هذا والجدول التالي يقدم لنا وصف مختصر للعناصر الموجودة في هذه الصفحة :

الوصف والاستخدام	العنصر
<p>عن طريق هذا العنصر يمكن إنشاء تطبيق يعمل بأى من خوادم شبكة الويب. وهذا الإنشاء يتم من خلال صندوق الحوار New Soap Server Application الموضح في الشكل التالي :</p>	Soap Server Application



عن طريق هذا العنصر يمكن إنشاء Module بيانات للتطبيقات التى تعمل بخوادم شبكة الويب. وهذا الإنشاء يتم من خلال المعالج Soap Data Module Wizard فى الشكل التالى :

Soap Server
Data Module

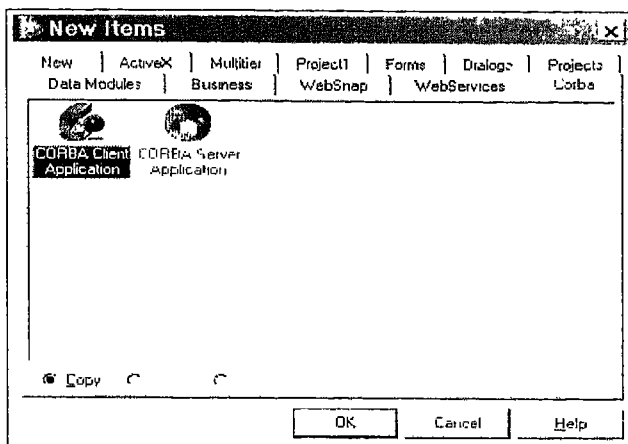
عن طريق هذا العنصر يمكن إنشاء أداة لاستيراد خدمات الويب وإضافتها للتطبيق الذى يعمل بخوادم شبكة الويب. وهذا الإنشاء يتم من خلال صندوق الحوار Web Services Import الموضح فى الشكل التالى :

Web Services
Importer



الصفحة CORBA بصندوق الحوار New Items

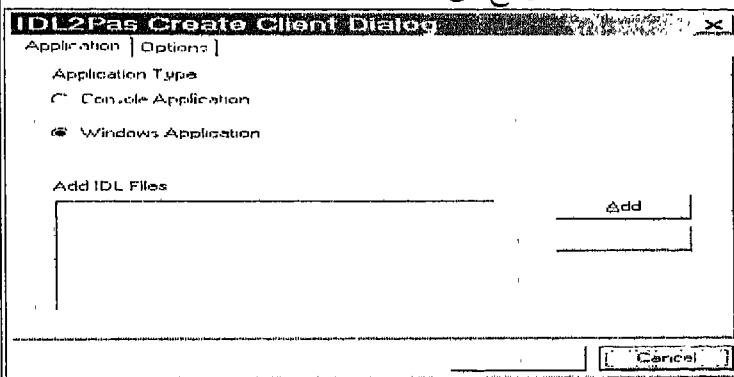
الشكل التالي يوضح لنا محتويات هذه الصفحة :

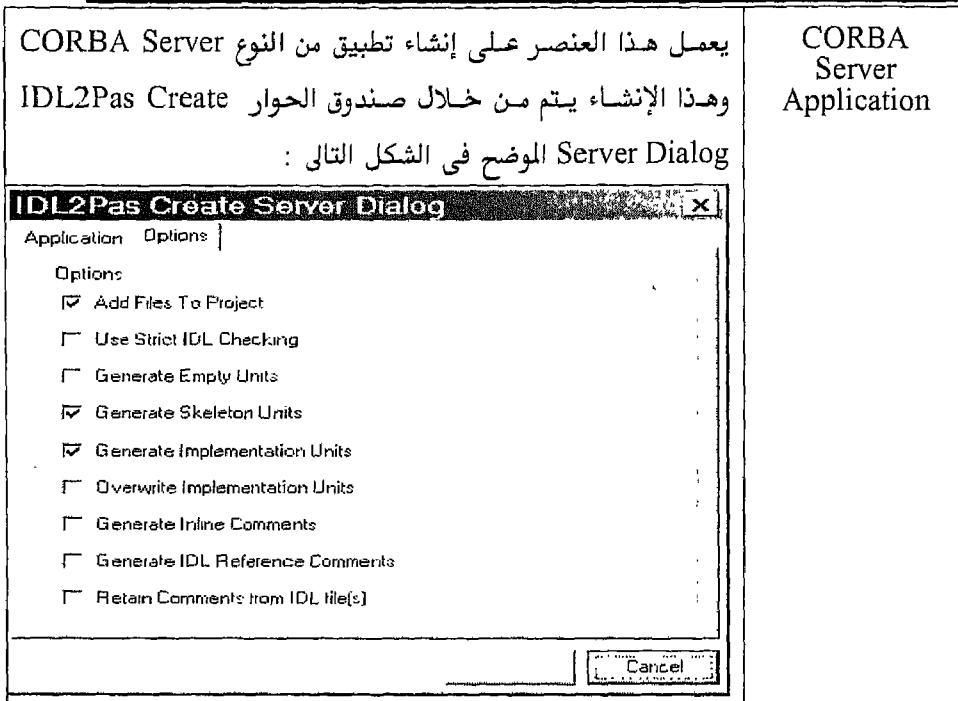


شكل توضيحي :

من خلال العناصر الموجودة في هذه الصفحة يمكن إنشاء تطبيقات CORBA سواء كانت للعملاء Client أو للخادم Server. هذا والجدول التالي يقدم لنا وصف مختصر للعناصر الموجودة في هذه الصفحة :

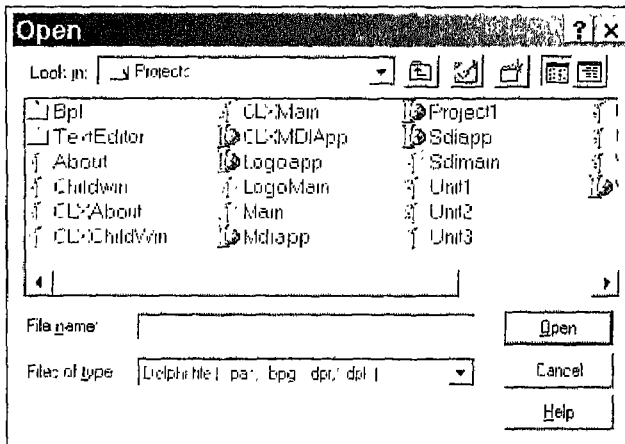
العنصر	الوصف والاستخدام
CORBA Client Application	يعمل هذا العنصر على إنشاء تطبيق من النوع CORBA Client Application و هذا الإنشاء يتم من خلال صندوق الحوار IDL2Pas Create Client Dialog الموضح في الشكل التالي :





الأمر Open بالقائمة File

يعمل هذا الأمر على عرض صندوق الحوار Open (الموضح فى الشكل التالى) من أجل فتح وتحميل مشروع أو فورمة أو وحدة تم إنشاؤها قبل ذلك كما يمكن أيضا فتح ملف نصي داخل محرر الكود البرمجي Code Editor :

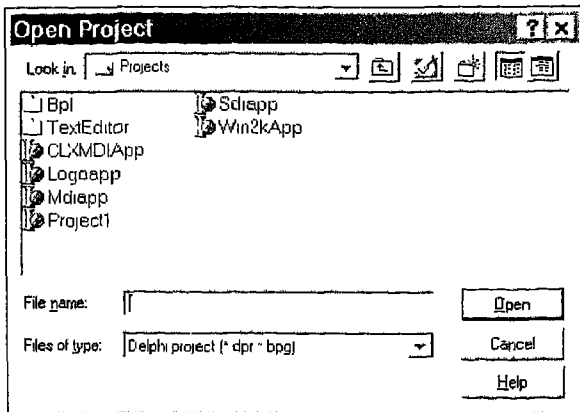


شكل توضيحي :



الأمر Open Project بالقائمة File

يعمل هذا الأمر على عرض صندوق الحوار Open Project (الموضح في الشكل التالي) والذي يمكن من خلاله فتح وتحميل أى مشروع من المشروعات التى سبق إنشاؤها (سواء كانت مخزنة فى ملف بالامتداد .BPK أو الامتداد .BPR) :



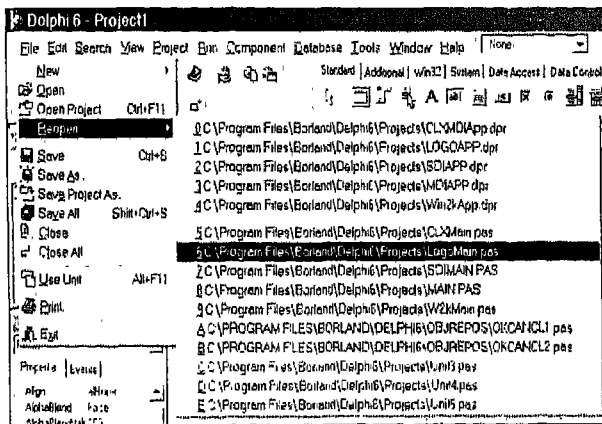
شكل توضيحي :

يمكن أيضا فتح صندوق الحوار Open Project عن طريق الضغط على المفاتيح **Ctrl+F11** بلوحة المفاتيح.



الأمر Reopen Project بالقائمة File

من خلال هذا الأمر يمكن إعادة فتح أى مشروع أو Module من المشاريع أو ال Modules التى سبق فتحها قريبا والتي يتم عرض أسماؤها ومواقعها فى القائمة الفرعية الخاصة به كما هو موضح بالشكل التالى :



شكل توضيحي :

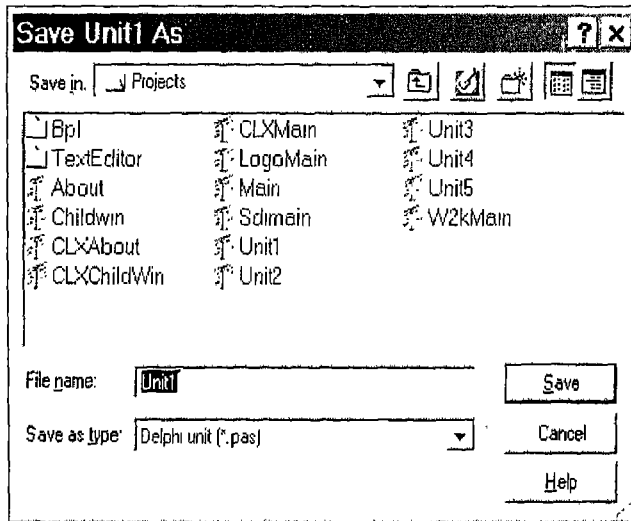
**الأمر Save بالقائمة File**

عن طريق هذا الأمر يتم حفظ الملف الذي يتم التعامل معه حاليا وهذا الحفظ يكون بنفس الأسم المخصص حاليا لهذا الملف

يمكن استدعاء الأمر Save عن طريق الضغط على المفاتيح Ctrl+S بلوحة المفاتيح.

**الأمر Save As بالقائمة File**

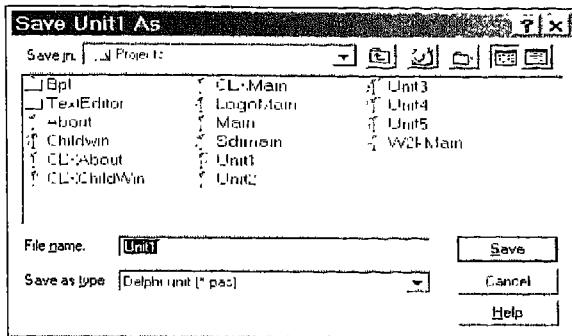
من خلال هذا الأمر يتم حفظ الملف الحالي باستخدام اسم جديد مما يؤدي إلى عمل نسخة من هذا الملف وهذه النسخة الجديدة تكون متضمنة كافة التعديلات التي تم إجروها مؤخرا على ملفات المشروع. وهذا الحفظ يتم من خلال صندوق الحوار Save As الموضح في الشكل التالي :



شكل توضيحي :

الأمر Save Project As بالقائمة File

يعمل هذا الأمر على حفظ المشروع الحالي بأسم جديد وذلك من خلال صندوق الحوار Save Project As الموضح بالشكل التالي :



شكل توضيحي :

الأمر Save All بالقائمة File

من خلال هذا الأمر يمكن حفظ كافة الملفات المفتوحة سواء كانت تخص المشروع الحالة أو Modules.

يمكن استدعاء الأمر Save All عن طريق الضغط على مجموعة المفاتيح Shift+ Ctrl+S بلوحة المفاتيح.



الأمر Close بالقائمة File

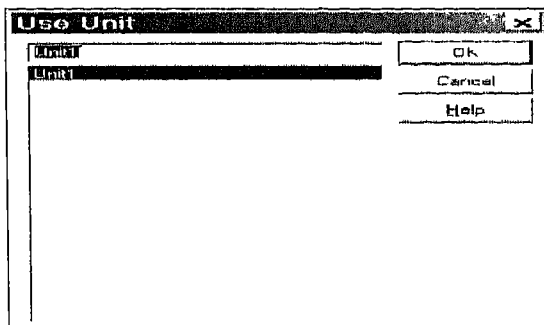
يعمل هذا الأمر على إغلاق المشروع الحالي وكافة الوحدات والفورم المرتبطة به.

الأمر Close All بالقائمة File

يعمل هذا الأمر على إغلاق كافة الملفات المفتوحة.

الأمر Use Unit بالقائمة File

يعمل هذا الأمر على إضافة الوحدة المختارة إلى الجملة uses الموجودة بالModule النشط حالياً وذلك من خلال صندوق الحوار Use Unit الموضح في الشكل التالي :



شكل توضيحي :

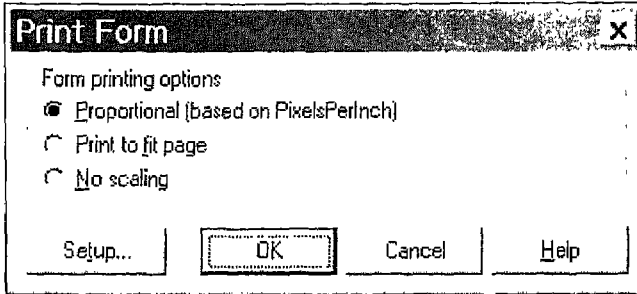


يمكن أيضا فتح صندوق الحوار Use Unit عن طريق الضغط على المفاتيح Alt+F11 بلوحة المفاتيح.



الأمر Print بالقائمة File

يقوم هذا الأمر بإرسال ملف الفورمة النشطة حاليا إلى الطابعة لطباعته وهذا الإرسال يتم من خلال صندوق الحوار Print Form الموضح فى الشكل التالى :



شكل توضيحي :

الأمر Exit بالقائمة File

يعمل هذا الأمر على غلق المشروع المفتوح وإنهاء التعامل مع لغة Delphi.

القائمة Edit

يتم استخدام مجموعة الأوامر الموجودة فى القائمة Edit للتعامل مع النصوص والمكونات فى مرحلة التصميم.

الجدول التالى يقدم لنا وصفا مختصرا للأوامر الموجودة بهذه القائمة :

الوصف والاستخدام	الأمر
هذا الأمر يلغى أثر آخر فعل قمت به كما يعيد العنصر الذى تم مسحه أخيرا.	Undo/Undelete
هذا الأمر يلغى تأثير الأمر Undo أو الأمر Undelete.	Redo
من خلال الأمر يتم استقطاع العنصر المختار من موضعه ويضعه فى لوحة الاقتباس clipboard.	Cut



يعمل هذا الأمر على وضع نسخة من العنصر المختار بلوحة الاقتباس وفى نفس الوقت يترك الأصل فى موضعه.	Copy
يعمل هذا الأمر على أخذ نسخة من محتويات لوحة الاقتباس ووضعه فى محرر نافذة الكود البرمجى Code Editor أو فى داخل الفورمة.	Paste
من خلال هذا الأمر يتم مسح العنصر المختار.	Delete
يؤدى هذا الأمر إلى اختيار كافة المكونات الموجودة فى الفورمة.	Select All
يعمل هذا الأمر على ضبط المكونات المختارة على أقرب نقطة فى شبكة النقط بالفورمة.	Align to Grid
من خلال هذا الأمر يتم نقل المكون المختار ليصبح فوق المكونات الأخرى وذلك فى حالة وجود مكونات بعضها فوق بعض.	Bring to Front
من خلال هذا الأمر يتم نقل المكون المختار ليصبح تحت المكونات الأخرى وذلك فى حالة وجود مكونات بعضها فوق بعض.	Send to Back
يعمل هذا الأمر على ضبط المكونات الموجودة فى الفورمة.	Align
عن طريق هذا الأمر يمكن تغيير حجم المكونات الموجودة فى الفورمة.	Size
يعمل هذا الأمر على تغيير حجم كافة المكونات الموجودة فى الفورمة.	Scale
من خلال هذا الأمر يتم التعديل فى ترتيب انتقال دفة التحكم focus بين المكونات (باستخدام المفتاح Tab الموجود بلوحة المفاتيح) الموجودة فى الفورمة النشطة.	Tab Order



Creation Order	يعمل هذا الأمر على تعديل الترتيب الذى تم من خلاله إنشاء المكونات الغير مرئية (فى مرحلة التشغيل Run-Time).
Flip Children	عن طريق هذا الأمر يتم عكس أو قلب تخطيط أدوات التحكم من اليمين إلى اليسار أو العكس كما لو كنا ننظر إليها فى مرآة.
Lock Controls	يعمل هذا الأمر على تأمين كافة المكونات الموجودة بالفورمة بحيث لا يمكن نقلهم من مواضعهم الحالية.
Add to Interface	من خلال هذا الأمر يمكن تعريف أسلوب أو حدث أو خاصية جديدة لأى أداة من أدوات التحكم ActiveX.

الأمر Undo/Delete بالقائمة Edit

فى أثناء العمل بمحرر الكود البرمجى Code Editor يمكن أن نختار الأمر Undo من القائمة Edit وذلك لإلغاء تأثير أحدث استخدام للوحة المفاتيح أو لأحدث فعل تم بالماوس. أما عند التعامل مع فورمة فإنه يتم اختيار الأمر Delete من القائمة Edit لاسترجاع آخر عنصر تم مسحه من الفورمة.

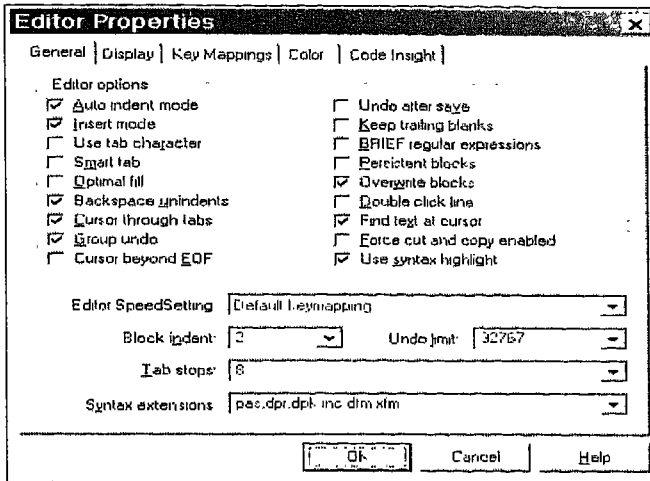
استخدام الأمر Undo فى محرر الكود البرمجى Code Editor

الأمر Undo يعمل على إعادة إدراج أى حروف قمت بمسحها كما يسمح أى حروف قمت بكتابتها كما يقوم أيضا باستبدال أى حروف قمت بتخطيها overwrite أو قد يعمل على جعل مؤشر الكتابة يعود مرة أخرى إلى موضعه السابق.

تستطيع ان تتراجع عن العديد من الأفعال التى قمت بها تباعا وذلك عن طريق اختيار الأمر Undo بصفة متكررة مما يؤدى إلى إزالة التغييرات التى أجريتها من خلال مبدأ "الرجوع للخلف تباعا" عبر الأفعال التى قمت بها والعودة مرة أخرى إلى الحالة السابقة.

لكى تقرر الحد الذى يقف عنده الأمر Undo :

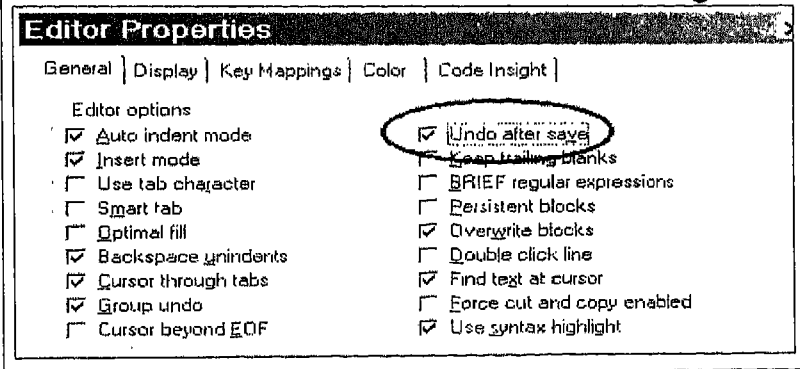
- (١) افتح القائمة Tools واختر منها الأمر Editor Options ليظهر على الشاشة صندوق الحوار Editor Options الموضح في الشكل التالي :



شكل توضيحي :

- (٢) انقر بالماوس على الحقل الخاص بالقائمة المنسدلة Undo limit وفيه اكتب القيمة العددية التي تمثل الحد الأقصى الذي ترغبه للأفعال التي يمكن التراجع عنها من خلال الأمر undo.

تستطيع أن تتراجع عن تغيير قمت به في الكود البرمجي بعد أن تحفظ المشروع وذلك في حالة أنك علمت بالماوس على الاختيار Undo after save في التبويب General بصندوق الحوار Editor Options كما هو موضح بالشكل التالي :





لو أنك تراجعت عن العملية التي تعرف بـ block operation في هذه الحالة يظهر الملف الذي تتعامل معه كما كان في السابق قبل أن تقوم بتنفيذ هذه العملية.

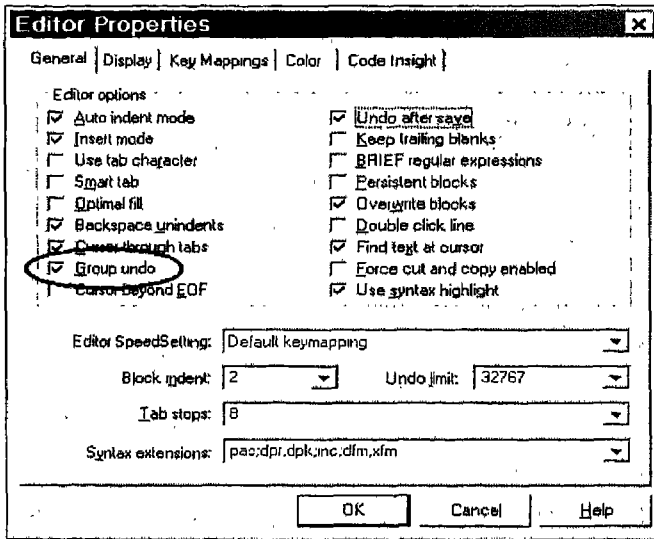
الأمر undo لا يعمل على تغيير اختيار تم تحديده ويكون له تأثير على أكثر من نافذة.



لكي تتراجع عن مجموعة من الأفعال قم بالآتي :

(١) افتح القائمة Tools واختر منها الأمر Editor options ليظهر على الشاشة صندوق الحوار Editor Options.

(٢) علم بالماوس على الاختيار Group Undo كما هو موضح بالشكل التالي :



شكل توضيحي :

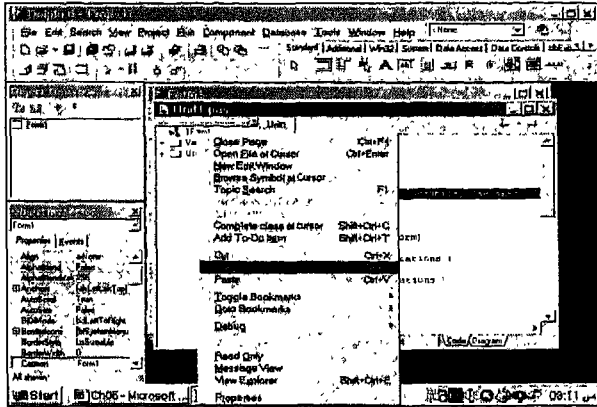
الأمر Redo بالقائمة Edit

يتم استخدام الأمر Redo من القائمة Edit لعكس التأثيرات التي نجمت عن آخر استخدام للأمر Undo. هذا والأمر Redo يكون له تأثير عندما يتم استخدامه بعد استخدام الأمر Undo مباشرة. هذا ويكون الأمر Redo غير متاح للاستخدام لعكس التأثيرات الناجمة عن الأمر Undelete.

الأمر Cut بالقائمة Edit (أو بالقائمة المختصرة بـ نافذ محرر الكود البرمجى)

اختيار الأمر Cut من القائمة Edit يؤدي إلى إزالة العناصر التالية من الموضع الحالية لها ووضعهم فى لوحة الاقتباس clipboard :

● النص المختار أى المعلم عليه فى محرر الكود البرمجى Code Editor (وأنت تستطيع أيضا النقر بالزر الأيمن للماوس على النص المعلم عليه ثم اختيار الأمر Cut من القائمة المختصرة التى تظهر على الشاشة كما هو موضح بالشكل التالى :



شكل توضيحي :

● المكونات المختارة فى الفورمة النشطة.

● القوائم المختارة فى صندوق الحوار Menu Designer.

هذا ويعمل الأمر Cut على استبدال المحتويات الحالية بلوحة الاقتباس بالعنصر المختار. وعلى العموم لكى تضع محتويات لوحة الاقتباس فى أى موضع ترغبه بالمشروع فى هذه الحالة افتح القائمة Edit ثم اختر منها الأمر Paste.

الأمر Copy بالقائمة Edit (أو بالقائمة المختصرة بـ نافذ محرر الكود البرمجى)

يمكن اختيار الأمر Copy من القائمة Edit (أو النقر بالزر الأيمن للماوس فى محرر الكود البرمجى Code Editor ثم اختيار الأمر Copy من القائمة المختصرة التى تظهر على الشاشة) وذلك لوضع نسخة طبق الأصل من النص المختار أو مكون مختار أو قائمة مختارة فى لوحة الاقتباس وترك الأصل بدون أن يحدث له أى تغيير. هذا والأمر Copy يستبدل المحتويات الحالية بلوحة الاقتباس بالعناصر المختارة.

هذا ولكى تضع محتويات لوحة الاقتباس بأى موضع عليك إذن أن تستخدم الأمر Paste من القائمة Edit.

**الأمر Paste بالقائمة Edit (أو بالقائمة المختصرة بنافذة محرر الكود البرمجى)**

يمكن اختيار الأمر Paste من القائمة Edit وذلك لإدراج محتويات لوحة الاقتباس داخل صفحة محرر الكود البرمجى Code Editor النشط أو الفورمة النشطة أو القائمة النشطة فى صندوق الحوار Menu Designer.

تستطيع أن تلتصق paste النصوص فقط فى نافذة محرر الكود البرمجى Code Editor أما المكونات فيتم لصقها فى الفورمة فقط فى حين أن عناصر القوائم يتم لصقها فى صندوق الحوار Menu Designer فقط.



عند إجراء عملية اللصق داخل نافذة محرر الكود البرمجى Code Editor فإنه يتم إدراج النص عند الموضع الموجود به مؤشر الكتابة. وأنت تستطيع النقر بالزر الأيمن للماوس فى محرر الكود البرمجى Code Editor ثم اختيار الأمر Paste من القائمة المختصرة التى تظهر على الشاشة.

عند إجراء عملية اللصق داخل فورمة فى هذه الحالة نجد أن المكونات الغير مرئية (فى مرحلة التشغيل Run-Time) يتم لصقها بالركن الأيسر العلوى للفورمة أما المكونات الظاهرة (فى مرحلة التشغيل Run-Time) فيتم لصقها فى نفس الموضع الذى تم فيه نسخ أو قص هذه المكونات.

عند إجراء عملية اللصق داخل صندوق الحوار Menu Designer فإنه يتم إدراج عناصر القوائم عند الموضع الموجود به مؤشر الماوس.

هذا وأنت تستطيع لصق المحتويات الحالية للوحة الاقتباس أى عدد من المرات حتى تقوم بعمل قص أو نسخ عنصر جديد داخل لوحة الاقتباس.

الأمر Delete بالقائمة Edit

يتم استخدام الأمر Delete من القائمة Edit لمسح النص المختار أو المكون المختار بدون وضع نسخه منه فى بوحة الاقتباس ومن ثم فإنك لا تستطيع لصق العنصر المسحول ولكن بالرغم من ذلك فإن تستطيع استعادته هذا العنصر مرة أخرى بأن تستخدم الأمر Undelete من القائمة Edit مباشرة بعد استخدام الأمر Delete من القائمة Edit.



المسح يكون مفيد لو أنك ترغب فى إزالة عنصر وفى نفس الوقت لا ترغب فى استبدال محتويات لوحة الاقتباس بهذا العنصر.

الأمر Select All بالقائمة Edit

يعمل الأمر Select All بالقائمة Edit على اختيار كل عنصر موجود فى النافذة النشطة. هذا وفى أداة تصميم الفورمة يتم استخدام الأمر Select All من القائمة Edit لاختيار كل مكون موجود فى الفورمة. وعندما تختار أكثر من مكون فى نفس الوقت تجد أن الخصائص المشتركة بين هذه المكونات هى التى تظهر فقط فى النافذة Object Inspector.

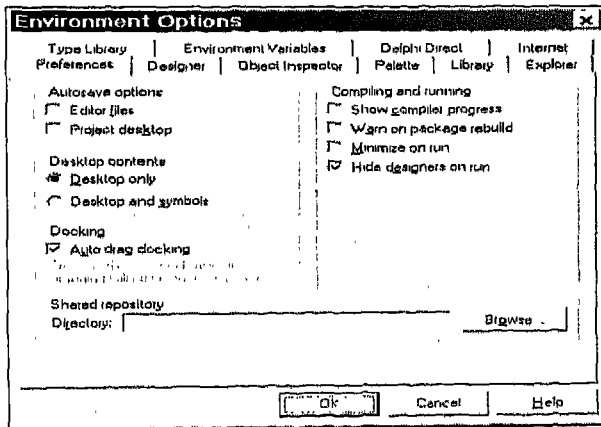
أما فى نافذة محرر الكود البرمجى Code Editor فإنه يتم استخدام الأمر Select All من القائمة Edit لاختيار كل النص الموجود فى الملف المعروض حالياً فى نافذة المحرر.

الأمر Align to Grid بالقائمة Edit (أو من القائمة المختصرة بالفورمة)

يتم اختيار الأمر Align to Grid من القائمة Edit لضبط المكونات المختارة على أقرب نقطة فى شبكة النقط الموجودة فى الفورمة بمود التصميم.

هذا ولكن تختار أكثر من مكون فى نفس الوقت فإما أن تضغط على المفتاح Shift بلوحة المفاتيح فى أثناء النقر بالماوس على كل مكون على حدة أو ترسم بالماوس مستطيل اختيار حول مجموعة المكونات التى ترغب فى اختيارها.

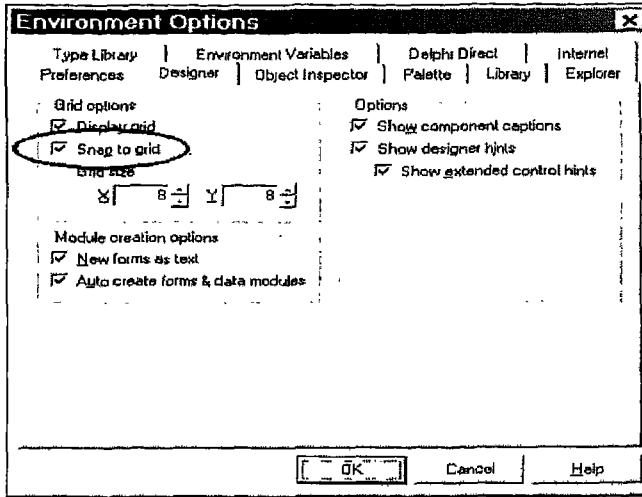
افتح القائمة Tools ثم اختر منها الأمر Environment Options ليظهر على الشاشة صندوق الحوار Environment Options الموضح فى الشكل التالى :



شكل توضيحي :



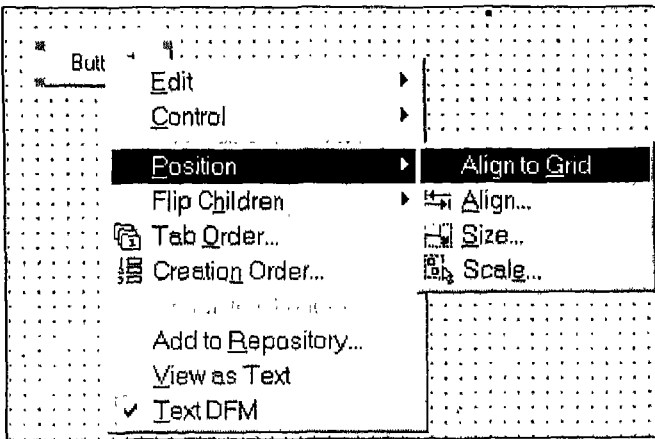
وفى التبويب Designer علم على الاختيار Snap to Grid كما هو موضح بالشكل التالى



شكل توضيحي :

ومن ثم فإن أى مكون سيضبط نفسه تلقائياً على نقط الشبكة عند إضافته للفورمة. كذلك تستطيع تحديد حجم شبكة النقط التى بالفوركة وذلك عن طريق كل من العداد X والعداد Y فى التبويب Designer بصندوق الحوار Environment Options.

تستطيع أن تصل إلى الأمر Align to Grid عن طريق الأمر Position الموجود بالقائمة المختصرة التى تظهر عند النقر بالزر الأيمن للماوس على أى موضع بالفورمة النشطة أو Module بيانات مع المكونات المختارة كما هو موضح بالشكل التالى :



شكل توضيحي :



الأمر Bring to Front بالقائمة Edit (أو بالقائمة المفتصرة بالفورمة)

من خلال الأمر Bring to Front الموجود بالقائمة Edit يتم نقل المكون المختار ليصبح فوق كافة المكونات الأخرى الموجودة فى الفورمة. ومثل هذا العمل يطلق عليه التغيير فى الترتيب العمودى للمكونات فى اتجاه محور Z.

لا يعمل الأمر Bring to Front فى حالة أنك اخترت خليط من المكونات (أدوات التحكم) بعضها له نافذة Windowed والآخر ليس له نافذة non-windowed. فعلى سبيل المثال لا تستطيع تغيير الترتيب العمودى Z-Order للكائن label فى أثناء اختياره مع الكائن button فى نفس الوقت.



الأمر Send to Back بالقائمة Edit (أو بالقائمة المفتصرة بالفورمة)

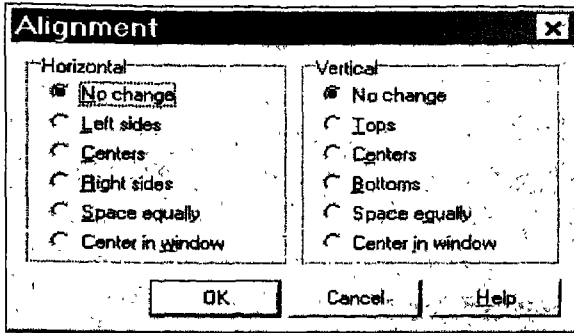
من خلال الأمر Send to Back الموجود بالقائمة Edit يتم نقل المكون المختار ليصبح تحت كافة المكونات الأخرى الموجودة فى الفورمة. ومثل هذا العمل يطلق عليه التغيير فى الترتيب العمودى للمكونات فى اتجاه محور Z.

لا يعمل الأمر Send to Back فى حالة أنك اخترت خليط من المكونات (أدوات التحكم) بعضها له نافذة Windowed والآخر ليس له نافذة non-windowed. فعلى سبيل المثال لا تستطيع تغيير الترتيب العمودى Z-Order للكائن label فى أثناء اختياره مع الكائن button فى نفس الوقت.



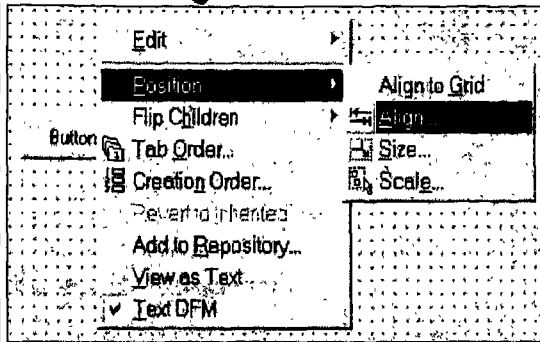
الأمر Align بالقائمة Edit

اختيار الأمر Align من القائمة Edit يؤدى إلى فتح صندوق الحوار Alignment الموضح فى الشكل التالى :



شكل توضيحي :

تستطيع أيضا الوصول للأمر Align (ولصندوق الحوار Alignment) عن طريق الأمر Position الموجود بالقائمة المختصرة التي تظهر عند النقر بالزر الأيمن للماوس بغورمة نشطة كما هو موضح بالشكل التالي :



صندوق حوار الضبط Alignment

يتم استخدام صندوق الحوار هذا من أجل عمل line up للمكونات المختارة في علاقة بين كل مكون والآخر أو في علاقة بينهم وبين الفورمة. هذا والإختيارات الخاصة بكل من الضبط الأفقي والضبط الرأسى عبارة عن الآتى :

الخيارات	الوصف والاستخدام
No Change	هذا الاختيار لا يفعل أى تغيير على ضبط المكون.
Left sides	يعمل هذا الاختيار على ضبط الحواف اليسرى للمكونات المختارة على خط واحد (فى الاتجاه الأفقى فقط).



يعمل هذا الاختيار على ضبط مراكز المكونات المختارة على خط واحد.	Centers
يعمل هذا الاختيار على ضبط الحواف اليمنى للمكونات المختارة على خط واحد (فى الاتجاه الأفقى فقط).	Right sides
يعمل هذا الاختيار على ضبط الحواف العليا للمكونات المختارة على خط واحد (فى الاتجاه الرأسى فقط).	Tops
يعمل هذا الاختيار على ضبط الحواف السفلية للمكونات المختارة على خط واحد (فى الاتجاه الرأسى فقط).	Bottoms
يعمل هذا الاختيار على ضبط المكونات المختارة على خط واحد بحيث تبعد عن بعضها البعض بمسافات متساوية.	Space Equally
يعمل هذا الاختيار على ضبط المكونات المختارة بالنسبة لمركز نافذة الفورمة.	Center in window

الأمر Size بالقائمة Edit

اختيار الأمر Size من القائمة Edit يؤدي إلى فتح صندوق الحوار Size الموضح فى

الشكل التالى :

شكل توضيحي :



صندوق الحوار Size

يمكن استخدام صندوق الحوار Size لتغيير حجم العديد من المكونات المختارة معا ليكونوا كلهم بنفس الارتفاع أو العرض بالضبط.

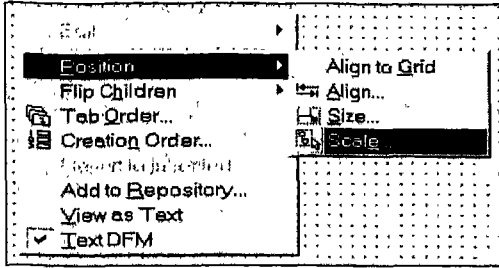
خيارات تحديد العرض Width تعمل على تغيير الحجم الأفقى للمكونات المختارة. أما خيارات تحديد الإرتفاع Height فتعمل على تغيير الحجم الرأسى للمكونات المختارة. هذا ويقدم لنا الجدول التالى الخيارات الخاصة بتحديد الحجم أفقيا أو رأسيا :

الاختيار	الوصف والاستخدام
No Change	هذا الاختيار لا يؤدي إلى حدوث أى تغيير على حجم المكونات المختارة.
Shrink to smallest	يعمل هذا الاختيار على تغيير حجم مجموعة من المكونات ليصبح ارتفاعها أو عرضها مساويا لارتفاع أو عرض أصغر مكون فى مجموعة المكونات.
Grow to largest	يعمل هذا الاختيار على تغيير حجم مجموعة من المكونات ليصبح ارتفاعها أو عرضها مساويا لارتفاع أو عرض أكبر مكون فى مجموعة المكونات.
Width	من خلال هذا الاختيار يمكن تحديد أى عرض نرغبه لمجموعة المكونات المختارة.
Height	من خلال هذا الاختيار يمكن تحديد أى ارتفاع نرغبه لمجموعة المكونات المختارة.



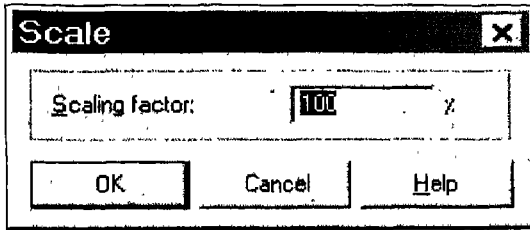
الأمر Scale بالقائمة Edit (أو بالقائمة المختصرة الخاصة بالفورمة)

اختيار الأمر Scale من القائمة Edit أو من القائمة الفرعية الخاصة بالأمر Position الموجود بالقائمة المختصرة التى تظهر عند النقر بالزر الأيمن للماوس على أى موضع بالفورمة كما هو موضح بالشكل التالى :



شكل توضيحي :

يؤدى ذلك إلى فتح صندوق الحوار Scale الموضح فى الشكل التالى :



شكل توضيحي :

صندوق الحوار Scale

يمكن استخدام صندوق الحوار Scale لتكبير أو تصغير حجم كافة المكونات الموجودة بالفورمة الحالية بنسبة معينة.

معامل التكبير والتصغير (نسبة مئوية)

فى الحقل المسمى Scaling Factor ادخل نسبة المئوية التى يتم من خلالها تكبير أو تصغير المكونات الموجودة بالفورمة علما بأنه إذا كانت القيمة أقل من ١٠٠٪ فإنها تؤدى إلى تصغير الحجم أما إذا كانت أكبر من ١٠٠٪ فإنها تعمل على تكبير الحجم.

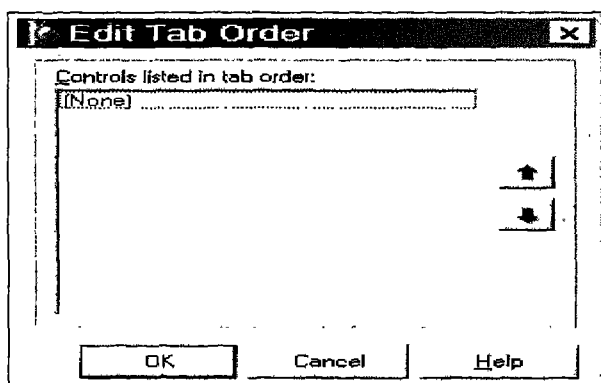
تتراوح قيمة المعامل Scaling Factor بين ٢٥٪ إلى ٤٠٠٪.





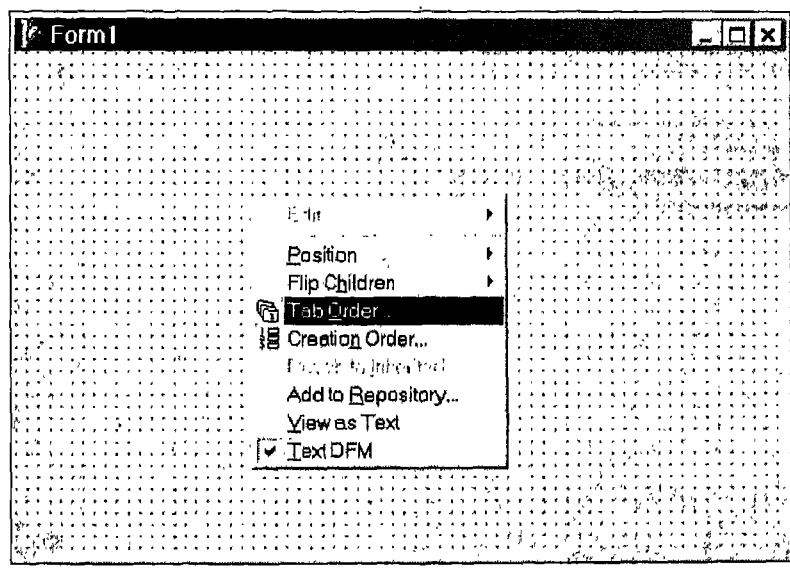
الأمر Tab Order بالقائمة Edit (أو بالقائمة المختصرة بالفورمة)

اختيار الأمر Tab Order من القائمة Edit يؤدي إلى فتح صندوق الحوار Edit Tab Order الموضح فى الشكل التالى :



شكل توضيحي :

تستطيع أيضا الوصول لصندوق الحوار Edit Tab Order عن طريق اختيار الأمر Tab Order من القائمة المختصرة التى تظهر عند النقر بالزر الأيمن للماوس على أى موضع بالفورمة كما هو موضح بالشكل التالى :





صندوق الحوار Edit Tab Order

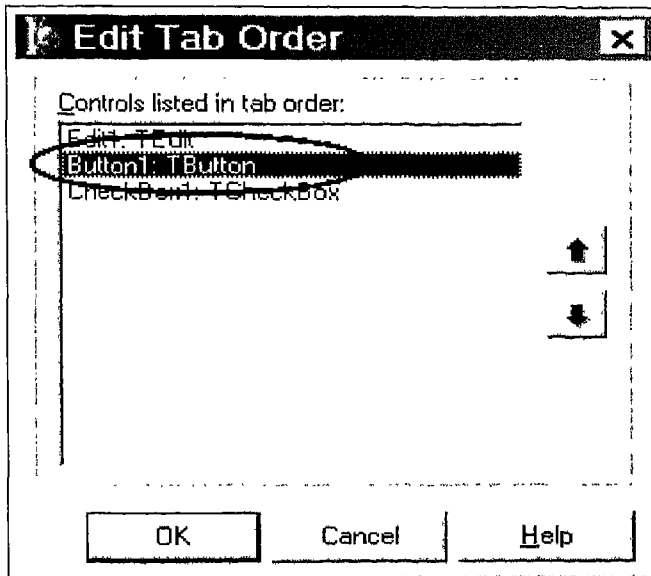
يمكن استخدام صندوق الحوار Edit Tab Order للتعديل فى ترتيب التنقل بين المكونات الموجودة بالفورمة (أو داخل المكون المختار فى حالة أنه يشتمل على مكونات أخرى داخله) باستخدام المفتاح Tab بلوحة المفاتيح فى مرحلة التشغيل Run-Time.

قائمة العرض Controls بصندوق الحوار Edit Tab Order

تشتمل قائمة العرض Controls بصندوق الحوار Edit Tab Order مجموعة المكونات الموجودة فى الفورمة النشطة بحيث أن هذه المكونات تظهر فى هذه القائمة بنفس ترتيب الوصول إليها بالفورمة. بمعنى أن أول مكون فى هذه القائمة هو أول مكون يتم التعامل معه فى مرحلة التشغيل Run-Time وهكذا... هذا ويتم تحديد ترتيب التنقل الافتراضى عن طريق الترتيب الذى تم به وضع المكونات فى الفورمة.

لكى تقوم بتغيير ترتيب التنقل الخاص بأى مكون اتبع الخطوات التالية :

- (١) اختر اسم المكون الذى ترغبه فى قائمة العرض Controls كما هو موضح بالشكل التالى :



شكل توضيحي :



(٢) انقر بالماوس على المفتاح Up  لنقل المكون المختار لأعلى فى قائمة العرض Controls أو انقر بالماوس على المفتاح Down  لنقله إلى أسفل فى قائمة العرض.

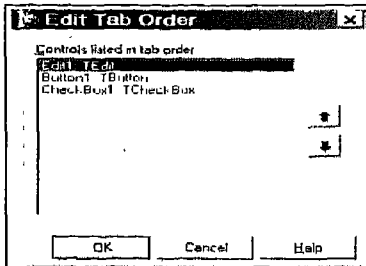
تستطيع أيضا سحب المكون المختار حتى تصل به إلى الموضع الذى ترغبه فى قائمة العرض Controls.



(٣) لحفظ التغييرات التى قمت به انقر بالماوس على المفتاح Ok.

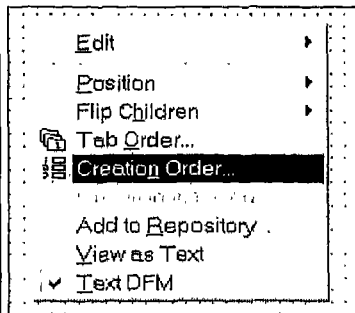
الأمر Creation Order بالقائمة Edit (بالقائمة المختصرة بالفورية)

اختيار الأمر Creation Order من القائمة Edit يؤدي إلى فتح صندوق الحوار Creation Order الموضح فى الشكل التالى :



شكل توضيحي :

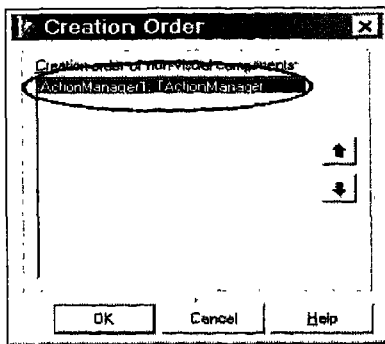
يمكنك أيضا الوصول إلى صندوق الحوار Creation Order عن طريق الأمر Creation Order الموجود بالقائمة المختصرة التى تظهر على الشاشة عند النقر بالزر الأيمن للماوس على أى موضع بالفورمة كما هو موضح بالشكل التالى :





صندوق الحوار Creation Order

يمكن استخدام صندوق الحوار Creation Order لتحديد ترتيب قيام التطبيق الذى تتولى إعداده بإنشاء المكونات الغير مرئية وذلك عندما تقوم بتحميل الفورمة التى تشتمل على هذه النوعية من المكونات سواء فى مرحلة التصميم أو فى مرحلة التشغيل Run-Time. قائمة العرض Creation order of non-visual components تكون مشتملة فقط المكونات الغير مرئية الموجودة فى الفورمة النشطة وهى تعرض نوع كل مكون وترتيب الإنشاء الحالى له كما هو موضح بالشكل التالى :

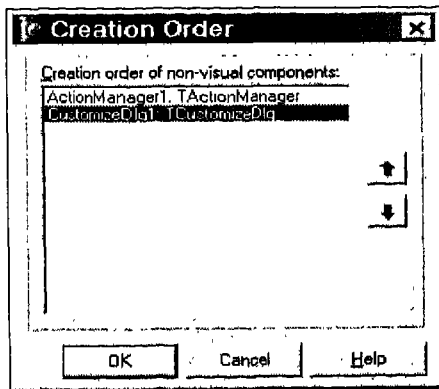


شكل توضيحي :



ففى هذا الصدد نقول إن ترتيب الإنشاء الافتراضى يتم تحديده عن طريق الترتيب الذى تم به وضع المكونات الغير مرئية فى الفورمة.

لكى تقوم بتغيير ترتيب الإنشاء لأى مكون من المكونات الغير مرئية اتبع الخطوات التالية :

- (١) اختر اسم المكون الذى ترغبه فى قائمة العرض Creation order of non-visual components كما هو موضح بالشكل التالى :



شكل توضيحي :

- (٢) انقر بالماوس على المفتاح  لنقل المكون المختار لأعلى فى قائمة العرض Creation order of non-visual components أو انقر بالماوس على المفتاح  لنقله إلى أسفل فى قائمة العرض.

تستطيع أيضا سحب المكون المختار حتى تصل به إلى الموضع الذى ترغبه فى قائمة العرض Creation order of non-visual components.

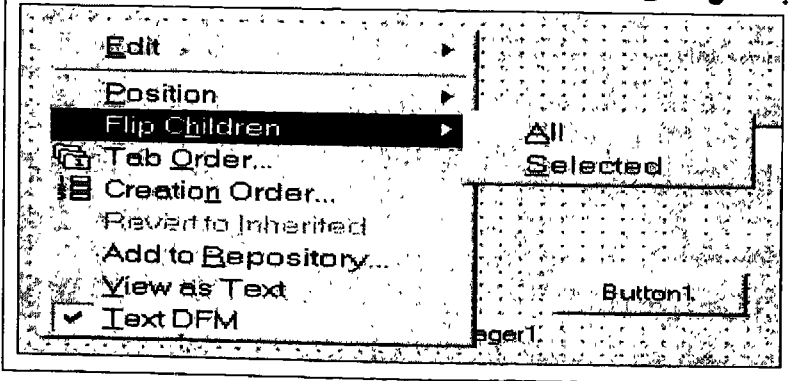


- (٣) لحفظ التغييرات التى قمت به انقر بالماوس على المفتاح Ok.

الأمر Flip Children بالقائمة Edit (بالقائمة المختصرة بالفورمة)

الأمر Flip Children الموجود فى القائمة Edit يسمح لك بأن تقلب أو تعكس تخطيط المكونات فى الفورمة الحالية لتصبح كما لو كنا ننظر إليها فى مرآة. ومثل هذا العمل يسمح للمبرمجين بأن يقوموا سريعا بتغيير أى فورمة من الوضع الطبيعى (من اليسار إلى اليمين) إلى الوضع المقابل أو المعاكس بحيث تصبح طبيعية بالنسبة لمستخدمى التطبيقات فى الوطن العربى.

يمكن الوصول لهذا الأمر عن طريق القائمة المختصرة التى تظهر عند النقر بالزر الأيمن للماوس على أى موضع بالفورمة كما هو موضح بالشكل التالى :





الأمر All بقائمة الأمر Flip Children بالقائمة Edit

يعمل هذا الأمر على قلب أو عكس مواضع كافة العناصر الموجودة فى الفورمة. كذلك فإنه يقلب اتجاه ضبط أى أدوات تحكم تم ضبطها إلى يسار أو إلى يمين الفورمة.

الأمر Selected بقائمة الأمر Flip Children بالقائمة Edit

يعمل هذا الأمر على قلب أو عكس مواضع كافة العناصر الوليدة children الموجودة فى المكونات. كما إنه يعمل أيضا على عكس اتجاه ضبط أى مكونات قد تم ضبطها إلى يمين أو يسار المكونات المختارة.

الأمر Lock Controls بالقائمة Edit

اختيار الأمر Lock Controls من القائمة Edit يؤدي إلى تأمين كافة المكونات الموجودة فى الفورمة النشطة مما يجعلها ثابتة فى مواضعها ومن ثم لا يمكن نقلها إلى مواضع أخرى بالفورمة. هذا وعندما يكون هذا الأمر معلم عليه فإنك لن تستطيع نقل أو تغيير حجم أى أداة من أدوات التحكم ولكن فى نفس الوقت تستطيع أن تستخدم النافذة Object Inspector.

الخصائص التى تستطيع التعامل معها من خلال النافذة Object Inspector لأدوات التحكم فى هذه الحالة عبارة عن الخاصية Height والخاصية Left والخاصية Top والخاصية Width.



عندما يكون الأمر Lock Controls معلم عليه فى القائمة Edit تكون أدوات التحكم مقفلة locked وعندما تكون أدوات التحكم مقفلة تستطيع حينئذ أن تختار مرة أخرى الأمر Lock Controls من القائمة Edit لإلغاء التعليم من عليه ومن ثم تصبح أدوات التحكم غير مقفلة.

الأمر Lock Controls ليس له أى تأثير على الفورمة نفسها. وعندما تختار الأمر Lock Controls فستظل لديك القدرة على تغيير حجم الفورمة أو نقلها لأى موضع ترغبه بالشاشة.



**الأمر Add to Interface بالقائمة Edit**

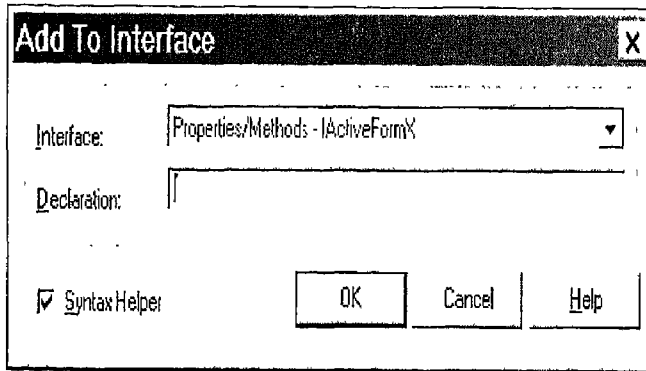
اختيار الأمر Add to Interface من القائمة Edit يعمل على تعريف إجراء جديد أو دالة جديدة أو خاصية جديدة لأى مكون من النوع ActiveX. وهذه العناصر يتم إضافتها إلى واجهات الاستخدام interfaces الخاصة بالمكون ActiveX مما يجعلهم متاحين للاستخدام بالنسبة للتطبيقات الأخرى. وهذا الأمر عبارة عن اختصار shortcut لعملية الإعلان عن عضو وسيط لكى يتم استخدامه بواسطة أى مكون من النوع ActiveX.

تستطيع أيضا النقر بالزر الأيمن للماوس على ملف التنفيذ الخاصة بالمكون ActiveX ثم تختار الأمر Add to Interface من القائمة المختصرة التى تظهر على الشاشة.



يعمل الأمر Add to Interface بالقائمة Edit على عرض صندوق الحوار

Interface الموضح فى الشكل التالى :



شكل توضيحي :

صندوق الحوار هذا يسمح لك بأن تختار نوعية العضو الوسيط (فقد يكون خاصية أو أسلوب أو حدث) وبعد ذلك تقوم سريعا بإدخال إعلان declaration لهذا النوع المختار. لعمل مراجعة تلقائية لصيغة ما تكتبه فى الحقل declaration فى هذه الحالة علم بالماوس على الاختيار Syntax Helper.

عندما تنقر بالماوس على المفتاح Ok يتم على الفور وبشكل تلقائى تخزين الإعلان الذى كتبته داخل المواضيع الأساسية التالية :



وحدة التنفيذ الحالية ActiveX Implementation.

مكتبة الأنواع ActiveX (المخزنة في ملف لت الامتداد TLB).

ملف الاستيراد Pascal importer (عبارة عن الملف _TLB.pas).

والآن يمكنك بكل بساطة كتابة كود برمجي بلغة الباسكال لهذا لأسلوب أو تحدد قيمة لهذه الخاصية وذلك في وحدة التنفيذ التي تتعامل معها.

القائمة Search

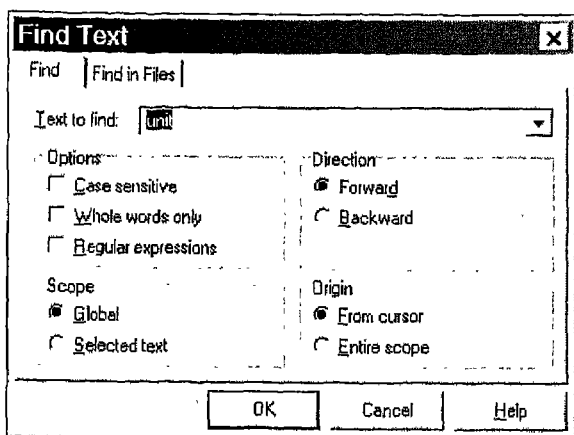
يمكن استخدام الأوامر الموجودة بالقائمة Search لكي تعثر على نصوص أو أخطاء أو كائنات أو وحدات أو متغيرات أو حتى رموز في محرر الكود البرمجي Code Editor. هذا والجدول التالي يقدم لنا وصف مختصر للأوامر الموجودة بهذه القائمة :

الوصف والاستخدام	الأمر
البحث عن نص معين والتعليم على أول نص يتم العثور عليه في محرر الكود البرمجي Code Editor.	Find
البحث عن نص معين وعرض كل نص يتم العثور عليه في نافذة تقع بالجزء السفلي بمحرر الكود البرمجي Code Editor.	Find in Files
البحث عن نص معين واستبداله بنص جديد.	Replace
إعادة آخر عملية بحث.	Search Again
البحث عن النص في أثناء كتابته.	Incremental Search
نقل مؤشر الكتابة لسطر معين في محرر الكود البرمجي Code Editor.	Go to Line Number
البحث عن آخر خطأ وقع في مرحلة التشغيل Run-Time.	Find Error
البحث عن رمز معين.	Browse Symbol



الأمر Find بالقائمة Search

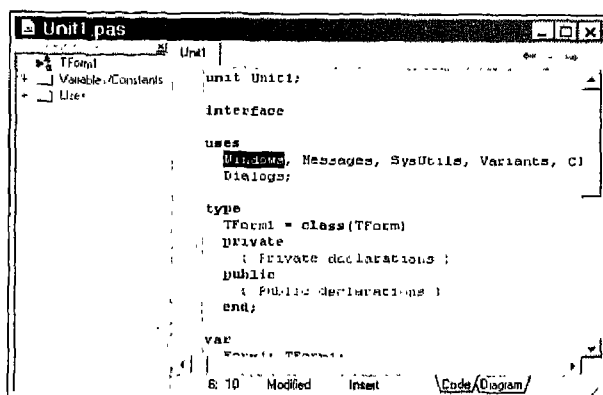
اختيار الأمر Find من القائمة Search يؤدي إلى فتح صندوق الحوار Find Text الموضح فى الشكل التالى :



شكل توضيحي :

صندوق الحوار Find Text

من خلال التبويب Find بصندوق الحوار Find Text تستطيع أن تحدد النص الذى ترغب فى العثور عليه كما تستطيع أيضا من خلال هذا التبويب تحديد الخيارات التى تؤثر فى عملية البحث. هذا وصندوق الحوار Find Text يحدد موضع السطر بالكود البرمجي الذى يشتمل على أول نص يتم العثور عليه كما يقوم بالتعليم على هذا النص فى نافذة محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالى :



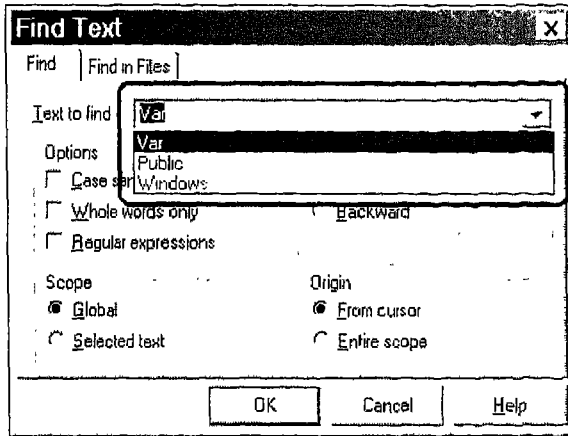
شكل توضيحي :



الخيارات الخاصة بصندوق الحوار Find Text

الحقل Text to find

ادخل السلسلة الحرفية المطلوب العثور عليها فى حقل القائمة المنسدلة Text to find او انقر بالماوس على السهم الموجود بهذه القائمة المنسدلة لفتحها ثم اختر نص من النصوص التى سبق البحث عنها كما هو موضح بالشكل التالى :



شكل توضيحي :

فيما يلى سنستعرض سويا الخيارات التى تحدد صفات عملية البحث التى توجد فى القسم Options بصندوق الحوار :

الاختيار	الوصف والاستخدام
Case sensitive	تمييز الحروف الكبيرة مثل ABC عن الحروف الصغيرة مثل a,b,c عند أداء عملية البحث.
Whole words only	البحث عن كلمات فقط. (ومن خلال كون هذا الاختيار off-أى لا يعمل- يتم البحث عن السلسلة الحرفية داخل سلاسل حرفية أكبر.
Regular expressions	الأخذ فى الاعتبار التعبيرات العادية فى أثناء البحث عن السلسلة الحرفية.



فيما يلي سنستعرض سويا الخيارات التى تحدد اتجاه البحث :

الاختبار	الوصف والاستخدام
Forward	جعل البحث يتم بداية من الموضع الموجود به مؤشر الكتابة حتى نهاية الملف.
Backward	جعل البحث يتم بداية من الموضع الموجود به مؤشر الكتابة حتى بداية الملف.

الاختبار Forward هو الاختيار الافتراضى.



فيما يلي سنستعرض سويا الاختيارات التى تحدد نطاق البحث داخل الملف :

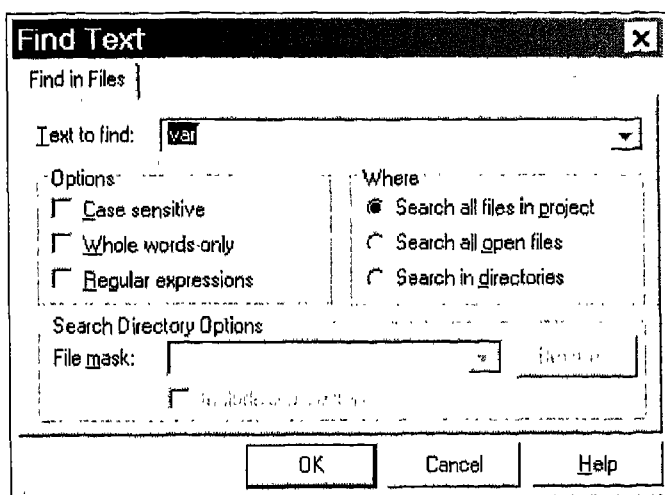
الاختبار	الوصف والاستخدام
Global	هذا الاختبار يجعل نطاق البحث عبارة عن الملف بأكمله وذلك فى الاتجاه المحدد بواسطة الخيارات التى تحدد إتجاه البحث.
Selected text	من خلال هذا الأمر يتم البحث فى النص المختار بالإتجاه المحدد بالخيارات التى تحدد إتجاه البحث. وأنت تستطيع استخدام الماوس أو أوامر البلوك لاختيار بلوك نصى.
From cursor	هذا الاختبار يجعل عملية البحث تبدأ من عند الموضع الحالى لمؤشر الكتابة وبعد ذلك تستمر إما للأمام حتى نهاية مدى البحث أو للخلف حتى بداية مدى البحث وهذا يتحدد بناء على الاختيارات التى تحدد اتجاه البحث.
Entire Scope	من خلال هذا الاختبار تغطى عملية البحث إما البلوك كله الموجود به النص المختار أو تغطى الملف بأكمله (لا يهم أين يوجد مؤشر الكتابة فى الملف) وهذا يتحدد بناء على الخيارات الموجودة فى القسم Scope بصندوق الحوار Find.

الاختيار Global هو الاختيار الافتراضى فى القسم Scope بصندوق الحوار Find.

الاختيار From cursor هو الاختيار الافتراضى بالقسم Origin فى صندوق الحوار Find.

الأمر Find in Files بالقائمة Search

اختيار الأمر Find in Files من القائمة Search يؤدى إلى فتح صندوق الحوار Find Text الموضح فى الشكل التالى :



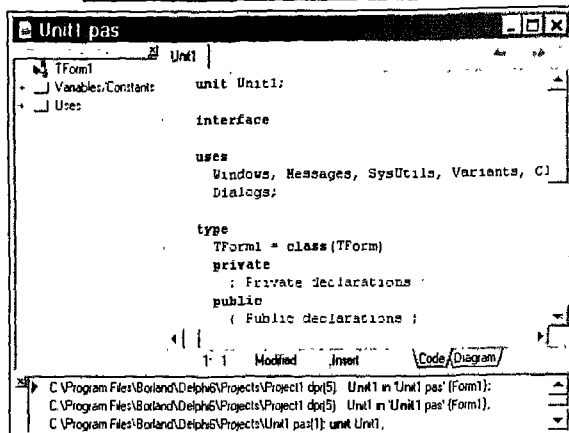
شكل توضيحي :

صندوق الحوار Find Text

صندوق الحوار Find Text يشتمل على تبويب واحد فقط وهو التبويب Find in Files وفى هذا التبويب يتم تحديد النص الذى ترغب فى تحديد موضعه كما يتم فيه تحديد الخيارات التى تؤثر فى عملية البحث. هذا وكل وجود للنص (السلسلة الحرفية) يتم سرده فى صندوق يوجد فى أسفل نافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

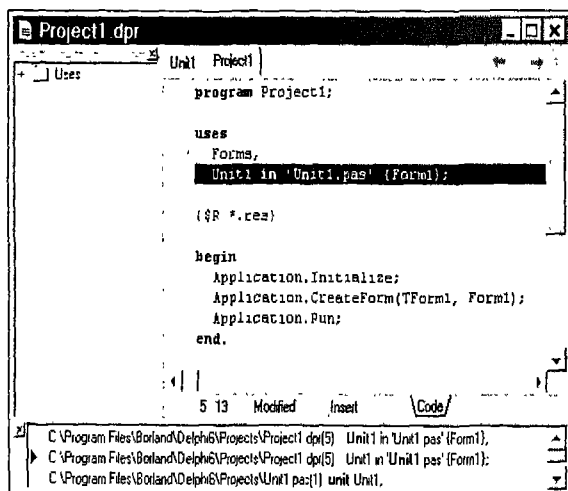


شكل توضيحي :



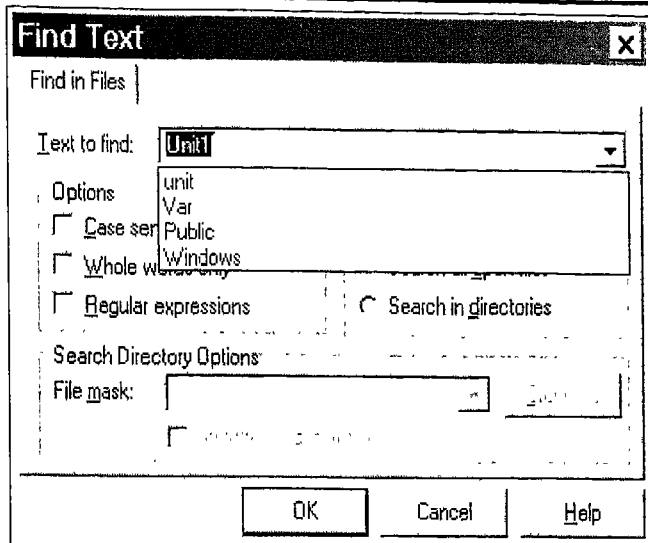
وعن طريق النقر بالماوس نقرا مزدوجا على أى عنصر من العناصر الموجودة فى هذا الصندوق يتم الإنتقال إلى السطر الموجود به هذا العنصر بملف الكود البرمجى كما هو موضح بالشكل التالى :

شكل توضيحي :



حقل القائمة المنسدلة Text to Find

فى حقل القائمة المنسدلة Text to Find تكتب السلسلة الحرفية التى ترغب فى البحث عنه. ولكى تختار من بين السلاسل الحرفية التى سبق البحث عنها فى هذه الحالة افتح هذه القائمة المنسدلة واختر منها ما ترغبه كما هو موضح بالشكل التالى :



شكل توضيحي :

الخيارات الموجودة فى القسم Options

الجدول التالى يقدم لنا وصف مختصر لوظيفة كل اختيار من الخيارات الموجودة

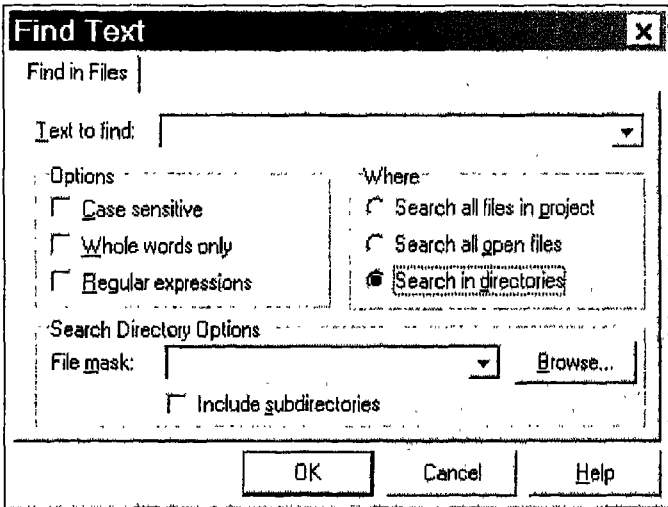
فى القسم Options :

الاختيار	الوصف والاستخدام
Case sensitive	تمييز الحروف الكبيرة مثل ABC عن الحروف الصغيرة مثل a,b,c عند أداء عملية البحث.
Whole words only	البحث عن كلمات فقط. (ومن خلال كون هذا الاختيار off-أى لا يعمل-يتم البحث عن السلسلة الحرفية داخل سلاسل حرفية أكبر.
Regular expressions	الأخذ فى الاعتبار التعبيرات العادية فى أثناء البحث عن السلسلة الحرفية.

فيما يلى سنستعرض سويا الخيارات الموجودة فى القسم Where بصندوق الحوار

Find in Files وذلك من خلال الجدول التالى :

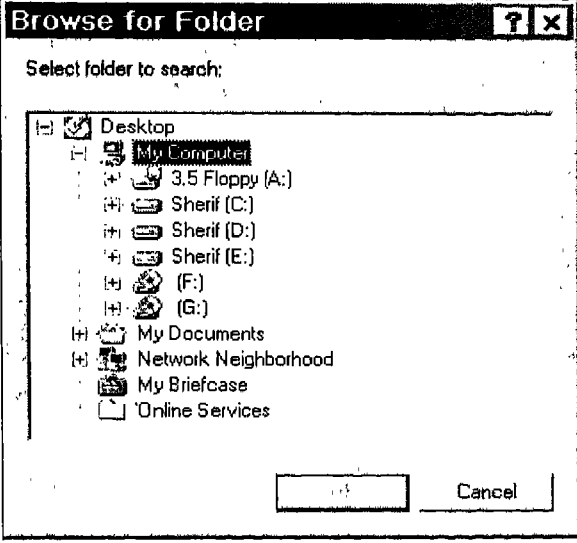


الوصف والاستخدام	الاختيار
من خلال هذا الاختيار يتم البحث فى كافة الملفات الموجودة فى المشروع المفتوح حاليا.	Search all files in project
من خلال هذا الاختيار يتم البحث فى كافة الملفات المفتوحة حاليا فقط.	Search all open files
<p>عند التعليم على هذا الاختيار تصبح الخيارات الموجودة فى القسم Search Directory Options متاحة للاستخدام كما هو موضح بالشكل التالى :</p>  <p>وفى هذه الحالة تتم عملية البحث عبر كافة الملفات الموجودة بالفهرس الذى يتم تحديده فى القائمة المنسدلة File mask.</p>	Search in directories

الاختيارات الموجودة بالقسم Search Directory Options

من خلال الجدول التالى نستعرض سويا الخيارات الموجودة فى القسم Search

: Directory Options

الوصف والاستخدام	الاختبار
<p>من خلال هذه القائمة المنسدلة يتم تحديد المسار الدال على مواقع الملفات التي سيتم البحث داخلها. ومن الطبيعي أن يتم البحث في الملفات التي لها الامتداد PAS. والامتداد DPR. فقط. أما لكي يتم البحث في ملفات أخرى في هذه الحالة استخدم حروف البحث التجميعية (مثل *.* أو *.txt) في نهاية المسار.</p> <p>لكي يتم البحث عن ملفات موجودة في الفهرس الأصلي الذي تم فيه تركيب لغة Delphi في هذه الحالة استخدم \$(DELPHI) لتحديد هذا الفهرس. فعلى سبيل المثال للبحث بكافة الملفات الموجودة في الفهرس Include في هذه الحالة استخدم \$(DELPHI)\Include أما لكي يتم البحث بالملفات الموجودة في المجلدات الموجودة بها أمثلة في هذه الحالة استخدم \$(DELPHI)\demos*.pas وهكذا...</p> <p>كذلك تستطيع أن تستخدم المفتاح Browse للتحديد المسار الذي ترغبه وذلك من خلال صندوق الحوار Browse for folders الموضح في الشكل التالي :</p>	<p>File mask</p>
	



لو أنك علمت بالماوس على هذا الاختيار فإن ذلك سيؤدي إلى البحث بالملفات الموجودة بالمجلدات الفرعية الموجودة بالمجلد المسار إليه بالمسار.	Include subdirectories
--	------------------------

إذا أردت أن تدخل أكثر من مسار بحث في القائمة المنسدلة File mask في هذه الحالة يجب أن تفصل بين كل مسار والذي يليه بالفاصلة المنقوطة.



الأمر Replace بالقائمة Search

اختيار الأمر Replace من القائمة Search يعمل على عرض صندوق الحوار Replace Text الموضح في الشكل التالي :

شكل توضيحي :

صندوق الحوار Replace Text

يمكن استخدام صندوق الحوار Replace Text لتحديد النص الذي ترغب في البحث عنه لاستبداله بنص آخر (أو بلا شيء).

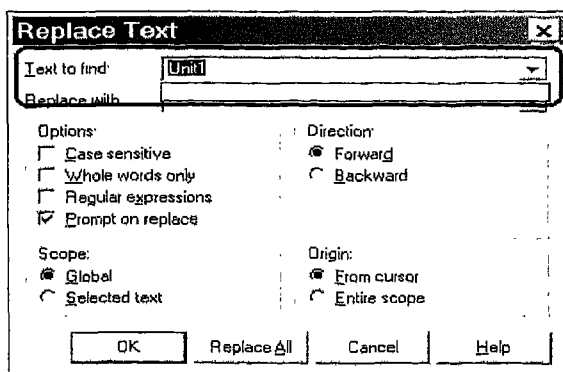
أغلب المكونات الموجودة في صندوق الحوار Replace Text تتطابق مع المكونات الموجودة في صندوق الحوار Find Text.





حقل القائمة المنسدلة Text to Find

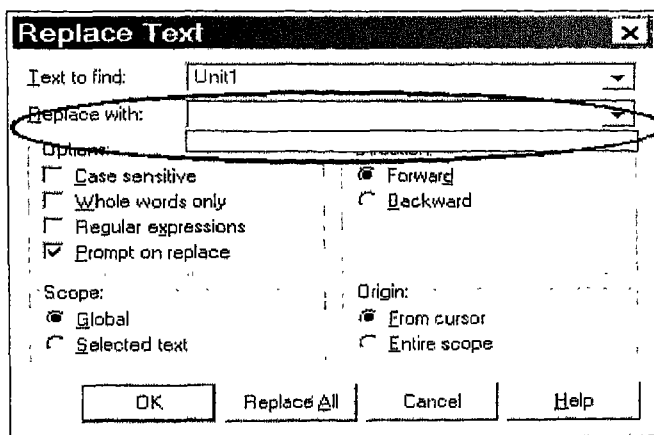
فى حقل القائمة المنسدلة Text to Find تكتب السلسلة الحرفية التى ترغب فى البحث عنه. ولكى تختار من بين السلاسل الحرفية التى سبق البحث عنها فى هذه الحالة افتح هذه القائمة المنسدلة واختر منها ما ترغبه كما هو موضح بالشكل التالى :



شكل توضيحي :

حقل القائمة المنسدلة Replace With

فى حقل القائمة المنسدلة Replace With تكتب السلسلة الحرفية التى سيتم استبدالها بالنص الموجود فى حقل القائمة المنسدلة Text to Find. ولكى تختار من بين السلاسل الحرفية التى سبق إدخالها فى هذه الحالة افتح هذه القائمة المنسدلة واختر منها ما ترغبه كما هو موضح بالشكل التالى :



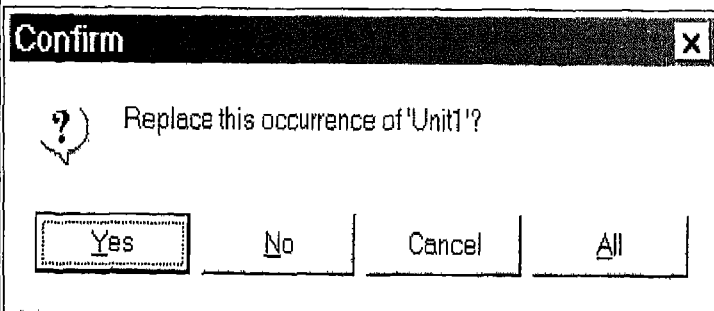
شكل توضيحي :



الخيارات الموجودة فى القسم Options

الجدول التالى يقدم لنا وصف مختصر لوظيفة كل اختيار من الخيارات الموجودة

فى القسم Options :

الاختيار	الوصف والاستخدام
Case sensitive	تمييز الحروف الكبيرة مثل ABC عن الحروف الصغيرة مثل a,b,c عند أداء عملية البحث.
Whole words only	البحث عن كلمات فقط. (ومن خلال كون هذا الاختيار off- أى لا يعمل- يتم البحث عن السلسلة الحرفية داخل سلاسل حرفية أكبر.
Regular expressions	الأخذ فى الاعتبار التعبيرات العادية فى أثناء البحث عن السلسلة الحرفية.
Prompt on replace	التعليم على هذا الاختيار يؤدي إلى ظهور الرسالة الموضحة فى الشكل التالى وذلك قبل إجراء عملية الاستبدال :  <p>وهذه الرسالة تطلب منك تأكيد لعملية الاستبدال. ولكن فى حالة عدم التعليم على هذا الاختيار فستتم عملية الاستبدال بدون أن تطلب منك تأكيد.</p>

فيما يلى سنستعرض سويًا الخيارات الموجودة فى القسم Direction والتي تحدد

اتجاه البحث :



الاختيار	الوصف والاستخدام
Forward	جعل البحث يتم بداية من الموضع الموجود به مؤشر الكتابة حتى نهاية الملف.
Backward	جعل البحث يتم بداية من الموضع الموجود به مؤشر الكتابة حتى بداية الملف.

فيما يلى سنستعرض سويا الاختيارات الموجودة فى القسم Scope والتي تحدد نطاق البحث داخل الملف :

الاختيار	الوصف والاستخدام
Global	هذا الاختيار يجعل نطاق البحث عبارة عن الملف بأكمله وذلك فى الاتجاه المحدد بواسطة الخيارات التى تحدد إتجاه البحث.
Selected text	من خلال هذا الأمر يتم البحث فى النص المختار بالإتجاه المحدد بالخيارات التى تحدد إتجاه البحث. وأنت تستطيع استخدام الماوس أو أوامر البلوك لاختيار بلوك نصي.

فيما يلى سنستعرض سويا الخيارات الموجودة فى القسم Origin والتي تحدد نقطة البداية :

الاختيار	الوصف والاستخدام
From cursor	هذا الاختيار يجعل عملية البحث تبدأ من الموضع الحالى لمؤشر الكتابة ثم تنطلق إما للأمام حتى الوصول لنهاية مدى أو نطاق البحث أو للخلف حتى الوصول لبداية مدى أو نطاق البحث.
Entire scope	من خلال هذا الاختيار يتم البحث فى كل البلوك الموجود به النص المختار أو فى الملف بأكمله (بغض النظر عن موضع مؤشر الكتابة بالملف) وذلك اعتمادا على الخيارات الموجودة فى القسم Scope.



المفتاح Replace All

النقر بالماوس على المفتاح Replace All يؤدي إلى استبدال كافة النصوص التي يتم العثور عليها (المطابقة للنص الموجود في حقل القائمة المنسدلة Text to find) بالنص الموجود في حقل القائمة المنسدلة Replace with. ولو كنت قد علمت على الاختيار Prompt on replace فسيؤدي ذلك إلى ظهور رسالة التأكيد Confirm قبل إجراء أى استبدال.

الأمر Search Again بالقائمة Search

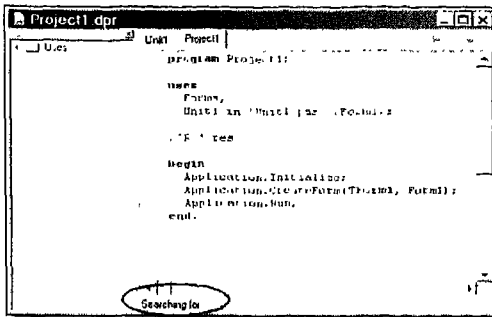
اختيار الأمر Search Again من القائمة Search يعمل على تكرار آخر عملية بحث أو استبدال. ونود هنا القول بأن القيم التحديدية التي تم إعدادها مؤخرا بصندوق الحوار Find Text أو بصندوق الحوار Replace Text تبقى كما هي وتؤثر نفس التأثير عندما تختار الأمر Search Again.

فعلى سبيل المثال لو أنك لم تقم بإلغاء التعليم من على الخيارات الموجودة في صندوق الحوار Replace Text في هذه الحالة سيقوم الأمر Search Again بالبحث عن آخر سلسلة حرفية ادخلتها في حقل القائمة المنسدلة Text to find ثم يستبدلها بالسلسلة الحرفية الموجودة في حقل القائمة المنسدلة Replace With.

الأمر Incremental Search بالقائمة Search

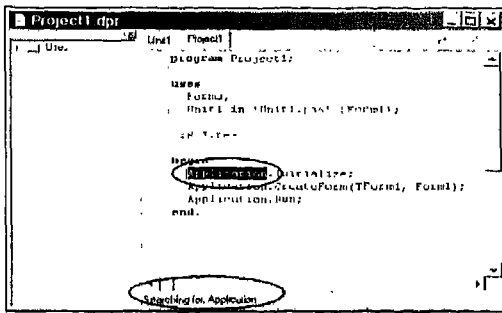
اختيار الأمر Incremental Search من القائمة Search يؤدي إلى تخطي bypass صندوق الحوار Find Text وذلك عن طريق نقل مؤشر الكتابة مباشرة إلى بداية النص الذي تكتبه بلوحة المفاتيح.

عندما تؤدي عملية بحث تزايدية ستلاحظ أن شريط الحالة لنافذة محرر الكود البرمجي Code Editor يكون مشتملا على الجملة Searching for كما إنه يعرض كل حرف تكتبه كما هو موضح بالشكل التالي :



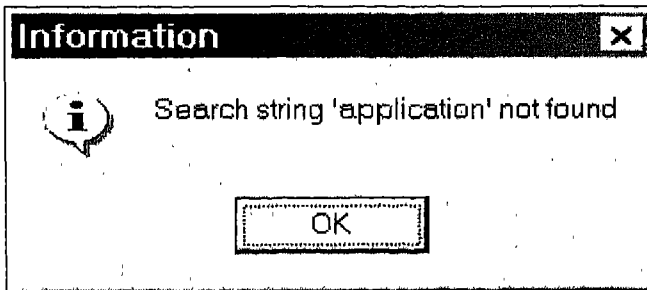
شكل توضيحي :

فعلى سبيل المثال لو أنك تكتب الكلمة Application ستجد أن مؤشر الماوس قد تحرك أى أول كلمة Application يتم العثور عليه كما يتم التعليم على كل حرف فيها فى أثناء الكتابة كما هو موضح بالشكل التالى :



شكل توضيحي :

ويستمر هذا السلوك حتى لا يكون هناك أى إحتمال للعثور على الكلمة المراد البحث عنا وفى هذه الحالة تظهر الرسالة الموضحة فى الشكل التالى :



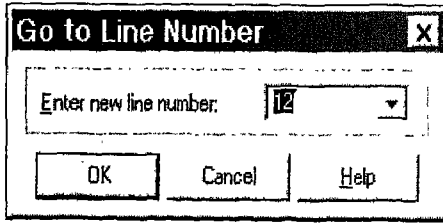
شكل توضيحي :

فيما يلى سنستعرض سويا من خلال الجدول التالى استخدامات بعض المفاتيح بلوحة المفاتيح فى أثناء إجراء عملية البحث التزايدية :

المفتاح	الاستخدام
Backspace	إزالة آخر حرف من السلسلة الحرفية والتحرك إلى آخر سلسلة حرفية تم العثور عليها سابقا.
F3	تكرار عملية البحث كما يمكن أيضا القيام بذلك من خلال الضغط على المفاتيح Ctrl+L بلوحة المفاتيح.

الأمر Go to Line Number بالقائمة Search

اختيار الأمر Go to Line Number من القائمة Search يؤدي إلى فتح صندوق الحوار Go To Line Number الموضح في الشكل التالي :



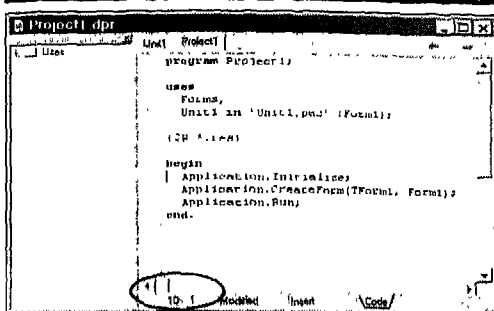
شكل توضيحي :

الضغط على المفاتيح Alt+G بلوحة المفاتيح يؤدي أيضا إلى فتح صندوق الحوار Go To Line Number.



صندوق الحوار Go To Line Number

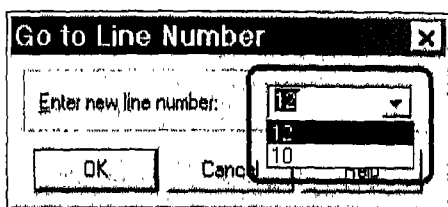
صندوق الحوار Go To Line Number يطلب منك أن تدخل رقم السطر الذي ترغب في العثور عليه بالكود البرمجي. هذا وستجد أن رقم السطر الموجود به مؤشر الكتابة حاليا معروضا في صندوق الحوار Go To Line Number عند عرضه على الشاشة. عند النقر بالماوس على المفتاح Ok يتم على الفور عرض كل من رقم السطر ورقم العمود الذي تم الوصول إليه وهذا العرض يكون في شريط الحالة لنافذة محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالي :



شكل توضيحي :

حقل القائمة المنسدلة Enter New Line Number

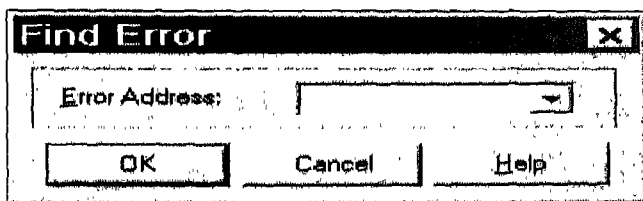
من خلال حقل القائمة المنسدلة Enter New Line Number يتم تحديد رقم السطر الذي ترغب في وضع مؤشر الكتابة عنده في نافذة محرر الكود البرمجي Code Editor. هذا ولكي تختار من أرقام السطور التي سبق إدخالها في هذه الحالة افتح القائمة المنسدلة واختر منها الرقم الذي ترغبه كما هو موضح بالشكل التالي :



شكل توضيحي :

الأمر Find Error بالقائمة Search

اختيار الأمر Find Error من القائمة Search يؤدي إلى فتح صندوق الحوار Find Error الموضح في الشكل التالي :



شكل توضيحي :

هذا الأمر يكون متاح للعمل فقط بعد أن يتم تشغيل التطبيق (وليس بعد أن يتم بناؤه فقط).



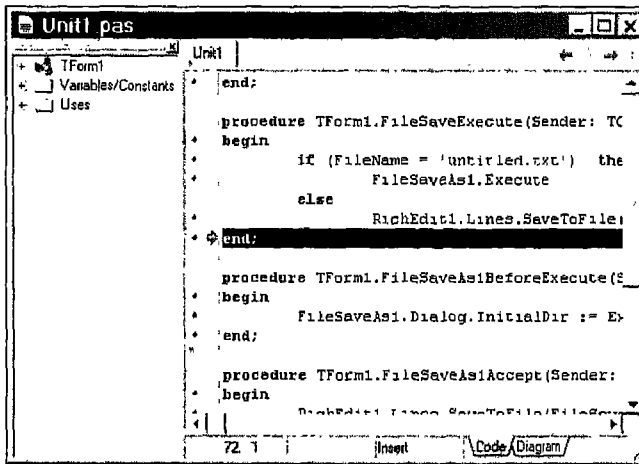


صندوق الحوار Find Error

يمكن استخدام صندوق الحوار Find Error لتحديد عنوان أحدث خطأ وقع فى مرحلة التشغيل Run-Time.

حقل القائمة المنسدلة Error Address

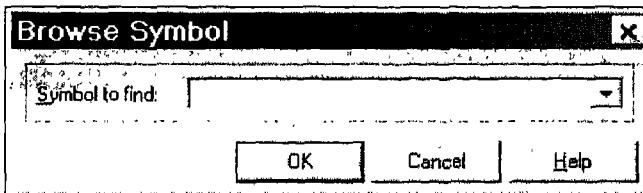
كل من عنوان أحدث خطأ تشغلي ورقم الخطأ يظهران فى تقرير مرحلة التشغيل Run-Time لو كانت هذه المعلومات متاحة. هذا وعندما تنقر بالماوس على المفتاح Ok تقوم لغة Delphi بإعادة ترجمة البرنامج الذى تعده وتقف عند عنوان الخطأ الذى أدخلته ويتم التعليم على الخطأ كما هو موضح بالشكل التالى :



شكل توضيحي :

الأمر Browse Symbol بالقائمة Search

اختيار الأمر Browse Symbol من القائمة Search يؤدي إلى فتح صندوق الحوار Browse Symbol الموضح فى الشكل التالى :



شكل توضيحي :

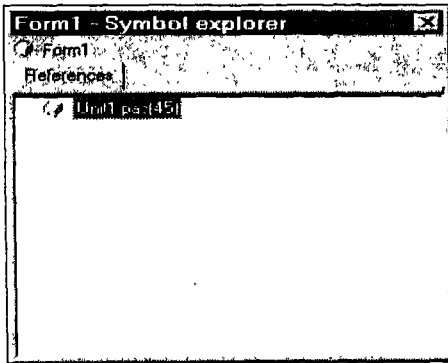
صندوق الحوار Browse Symbol

يتم استخدام صندوق الحوار Browse Symbol لعرض رمز معين. هذا وللقيام بذلك يمكنك القيام بأى من الآتى :

• كتابة اسم الرمز فى حقل القائمة المنسدلة Symbol to find ثم النقر بالماوس على المفتاح Ok.

• فتح القائمة المنسدلة Symbol to find واختيار واحد من الرموز التى سبق التعامل معها بصندوق الحوار ثم النقر بالماوس على المفتاح Ok.

عند النقر بالماوس على المفتاح Ok تقوم لغة Delphi بعرض معلومات عن الرمز الموصف فى صندوق الحوار Browse Symbol وهذه المعلومات تظهر فى نافذة تسمى Symbol Explorer. وهذه النافذة تعمل على توفير معلومات عن الإشارات المرجعية الموجود بالكود البرمجى للرمز المختار كما هو موضح بالشكل التالى :

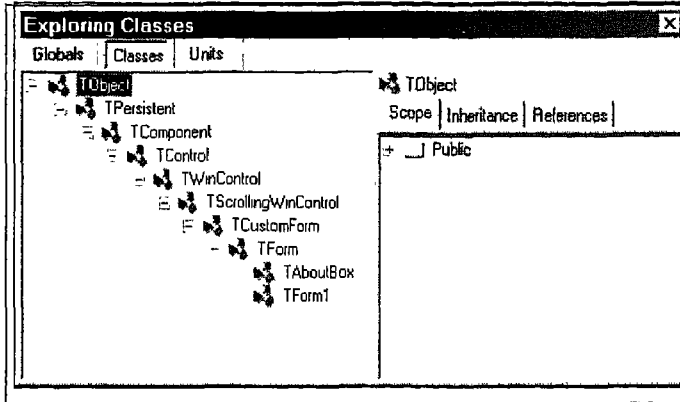


شكل توضيحي :

إذا كان الرمز عبارة عن قطاع class فى هذه الحالة نجد أن النافذة Symbol Explorer تعمل أيضا على توفير معلومات عن أعضاء هذا القطاع.

النافذة Symbol Explorer تشبه إلى حد كبير الجزء الأيمن للنافذة Project Browser كما هو موضح بالشكل التالى :



**القائمة View**

يمكن استخدام الأوامر الموجودة في القائمة View لعرض أو إخفاء مختلف العناصر التي تتألف منها بيئة التطوير المتكاملة IDE للغة Delphi كما يمكن أيضا فتح النوافذ التي تنتمي إلى الأداة المتكاملة لمعالجة الأخطاء.

الجدول التالي يقدم لنا وصف مختصر للأوامر الموجودة بالقائمة View :

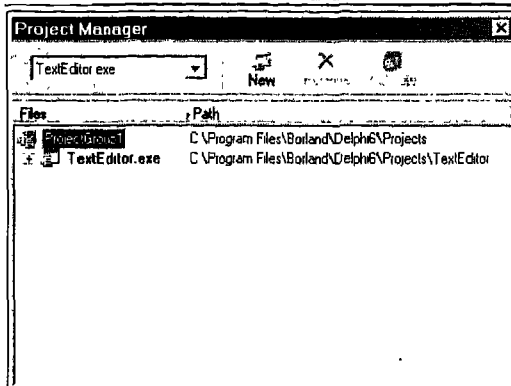
الأمر	الاستخدام
Project Manager	عرض مدير المشروع Project Manager على الشاشة.
Translation Manager	عرض مدير الترجمة Translation Manager على الشاشة
Object Inspector	عرض النافذة Object Inspector على الشاشة.
Object TreeView	استعراض الكائنات الموجودة في فورمة من خلال مشهد شجري.
To-Do List	هذا الأمر يسمح لك بأن تشاهد قائمة الواجبات الملحقة بالمشروع الذي يتم التعامل معها.
Alignment Palette	عرض بالليته الضبط على الشاشة.
Browser	عرض نافذة مستعرض المشروع Project Browser.



عرض نافذة متصفح الكود البرمجى Code Explorer.	Code Explorer
فتح صندوق حوار المكونات Components.	Component List
عرض قائمة تضم اسماء كافة النوافذ المفتوحة حاليا بالمشروع.	Window List
عرض القائمة الفرعية التى تشتمل على الأوامر الخاصة بأداة اكتشاف الأخطاء ومعالجتها Debugger.	Debug Windows
هذا الأمر يسمح لك بأن تعرض أو حفظ أو مسح المشاهد المتعددة والمختلفة لسطح المكتب.	Desktops
من خلال هذا الأمر يتم التنقل من الفورمة إلى نافذة الوحدة والعكس.	Toggle Form/Unit
فتح صندوق الحوار View Unit.	Units
فتح صندوق الحوار View Form.	Forms
عرض نافذة محرر مكتبة النوع Type Library.	Type Library
فتح محرر كود برمجى Code Editor جديد.	New Edit Window
إخفاء أو إظهار شرائط الأدوات أو بالليته المكونات.	Toolbars

الأمر Project Manager بالقائمة View

اختيار الأمر Project Manager من القائمة View يؤدي إلى فتح نافذة مدير المشروع Project Manager الموضحة فى الشكل التالى :



شكل توضيحي :

إذا كانت نافذة مدير المشروع Project Manager مفتوحة بالفعل فى هذه الحالة تصبح النافذة النشطة.



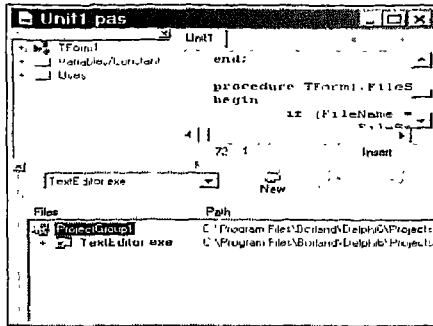
هناك طريقة أخرى لفتح نافذة مدير المشروع Project Manager وهى الضغط على مجموعة المفاتيح Ctrl+Alt+F11 بلوحة المفاتيح.



يمكن استخدام نافذة مدير المشروع Project Manager لمشاهدة مجموعة مشروع ومشاهدة المشروعات الموجودة فى مجموعة مشروع بالإضافة إلى إمكانية التجول بين الملفات الخاصة بالمشروع. وأنت تستطيع استخدام نافذة مدير المشروع Project Manager لإضافة مشاريع إلى مجموعة المشروع أو لمسح مشروعات منها أو لتنشيط (تفعيل) مشروع لو أن مجموعة المشروع التى تتعامل معها كانت تتألف من أكثر من مشروع.

كذلك تستطيع استخدام نافذة مدير المشروع Project Manager لإضافة أو مسح أو حفظ أو نسخ ملف إلى المشروع الذى يتم التعامل معه حالياً. وفى هذا الصدد نقول إن نافذة مدير المشروع Project Manager تعرض كافة الوحدات والفورمة المرتبطة بها والتى توجد فى المشروعات الموجودة بدورها داخل مجموعة المشروع التى يتم التعامل معها حالياً.

هذا وأنت تستطيع وضع نافذة مدير المشروع Project Manager فى أى مكان بسطح المكتب. كما إنك تستطيع أيضاً جعلها راسية docking أو ملتصقة مع نوافذ أخرى مثل نافذة محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالى :



شكل توضيحي :

فى هذا الشكل نشاهد نافذة مدير المشروع Project Manager وهى ملتصقة بالجزء السفلى بمحرر الكود البرمجى Code Editor.

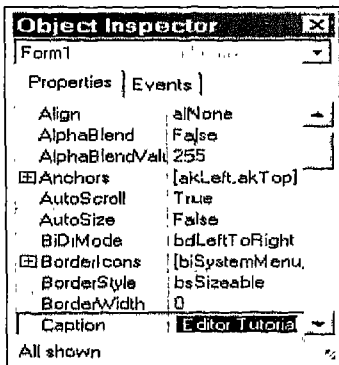


الأمر Translation Manager بالقائمة View

اختيار الأمر Translation Manager من القائمة View يؤدي إلى فتح نافذة مدير الترجمة Translation Manager. أما إذا كانت هذه النافذة مفتوحة بالفعل فى هذه الحالة تصبح النافذة النشطة.

الأمر Object Inspector بالقائمة View

لو أنك قمت بإغلاق النافذة Object Inspector فإن اختيار الأمر Object Inspector من القائمة View يؤدي إلى إعادة فتحها مرة أخرى وهذه النافذة نشاهدها فى الشكل التالى :



شكل توضيحي :

هناك طريقة أخرى لفتح النافذة Object Inspector وهى الضغط على المفتاح F11 بلوحة المفاتيح.

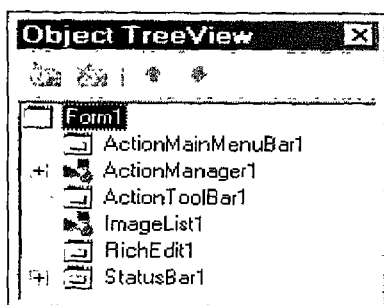




كذلك تستطيع اختيار الأمر Object Inspector من القائمة View للتنقل بين النافذة Object Inspector وآخر نافذة كانت نشطة أو نافذة محرر الكود البرمجى Code Editor. هذا ويتم استخدام النافذة Object Inspector للتعديل فى قيم الخصائص والوصلات الخاصة بأدوات معاملة الأحداث.

الأمر Object TreeView بالقائمة View

اختيار الأمر Object TreeView من القائمة View يؤدي إلى عرض النافذة Object TreeView على الشاشة فوق النافذة Object Inspector والشكل التالى يوضح لنا النافذة Object TreeView :



شكل توضيحى :

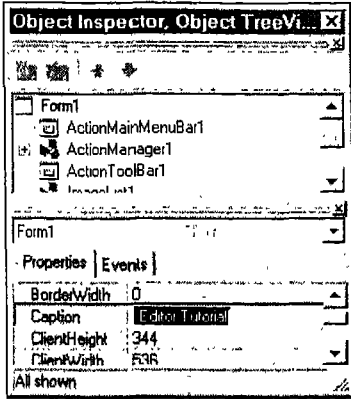
هناك طريقة أخرى لفتح النافذة Object Inspector وهى الضغط على مجموعة المفاتيح Shift+Alt+F11 بلوحة المفاتيح.



النافذة Object TreeView تعرض المكون المالك والمكونات الأخرى التابعة له (التي يطلق عليها أطفال). فعلى سبيل المثال تعتبر الفورمة هى المالك لأى مفتاح Button موجود داخلها وفى نفس الوقت نقول أن المفاتيح تعتبر أطفال الفورمة وهكذا... وفى هذا الصدد نقول إن هناك علاقة تبادلية مباشرة بين النافذة Object TreeView والنافذة Object Inspector ومن ثم فإن المشهد الشجرى يتغير عند حدوث أى تغيير فى النافذة Object Inspector والعكس صحيح أيضا.

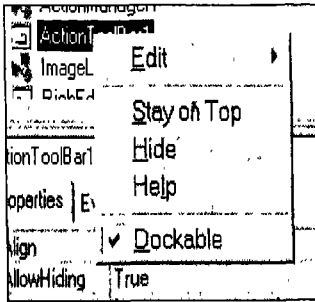
تمتلك النافذة Object TreeView كافة القدرات والإمكانات الخاصة بأداة تصميم Module البيانات. ومن بين هذه الإمكانيات إمكانية سحب واسقاط العناصر فى القوائم.

هذا وتعتبر النافذة Object TreeView من النوافذ القابلة للإرساء dockable بحيث يمكن لصقها فى النافذة Object Inspector كما هو موضح بالشكل التالى :







شكل توضيحي :

النقر بالزر الأيمن للماوس على أى عنصر من العناصر المعروضة فى النافذة Object TreeView يؤدى إلى ظهور قائمة مختصرة تشتمل على نفس الخيارات التى توجد فى القائمة المختصرة التى تظهر عند النقر بالزر الأيمن للماوس داخل نافذة المحرر Data Module كما هو موضح بالشكل التالى :



شكل توضيحي :

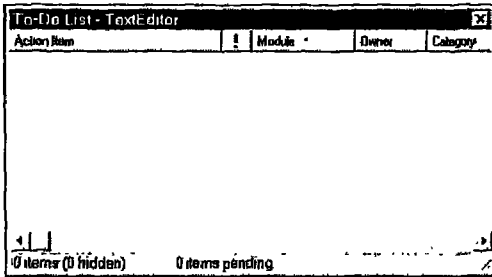
شريط الأدوات الخاص بالنافذة Object TreeView تشتمل على كل من الأيقونة New Item  والأيقونة Delete  والأيقونة Move Up  والأيقونة Move Down  . ولو أن هناك أكثر من نوع من العناصر التى يمكن إضافتها فى هذه الحالة ستجد أن الأيقونة New Item قد أصبح لها سهم صغير يشير لأسفل وعند النقر بالماوس على هذا السهم يؤدى إلى فتح قائمة صغيرة تشتمل على الأنواع المتاحة من العناصر التى يمكن إضافتها.



عندما يكون هناك كائن غير مكتمل فى هذه الحالة يتم عرضه فى النافذة Object TreeView وبجواره شاهد علامة صح حمراء. أما الأيقونات الشبحية Ghost فإنها تمثل مكونات تم إنشاؤها ضمنيا لكى يتم استخدامها فى الخلفية بدون أن يشعر بها المستخدم مثل المكون Default Session.

الأمر To-Do List بالقائمة View

اختيار الأمر To-Do List من القائمة View يؤدي إلى عرض النافذة To-Do List الموضحة فى الشكل التالى :



شكل توضيحي :

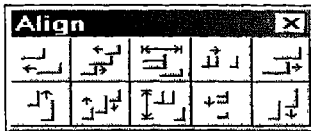
وهذه النافذة تكون خاصة بالمشروع الذى يتم التعامل معها حاليا. وهذه النافذة تعرض المهام التى ينبغى إنجازها لإستكمال المشروع الحالي.

العناصر الخاصة بالمشروع ككل يتم عرضها فى النافذة To-Do List. أما العناصر الموجودة فى المشروع والتى تمتلك كود برمجى غير مفتوح فى محرر الكود البرمجى Code Editor فإنها تظهر باللون الرمادى فى هذه النافذة.

تستطيع أن تقوم بفرز وترتيب العناصر الموجودة فى النافذة To-Do List وهذا الترتيب يمكن أن يكون عن طريق الأسم أو عن طريق الحالة أو عن طريق الأهمية والأولوية وهذا الترتيب يتم عن طريق النقر بالماوس على رأس العمود المناسب.

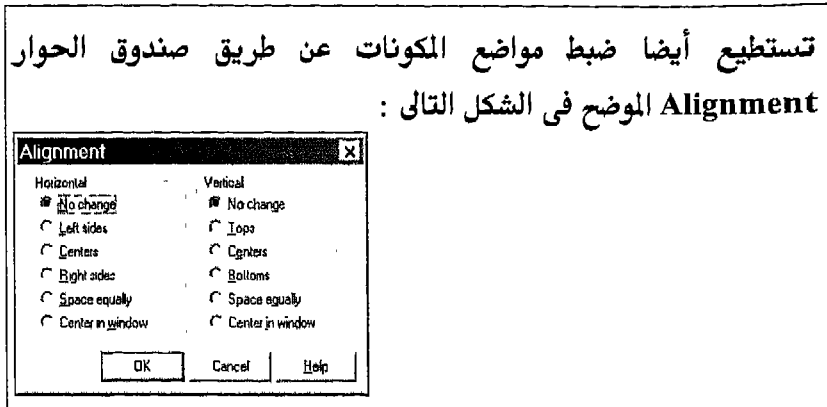
الأمر Alignment Palette بالقائمة View

اختيار الأمر Alignment Palette من القائمة View يؤدي إلى عرض باليئة الضبط الموضحة فى الشكل التالى :



شكل توضيحي :

من خلال هذه البالته يمكنك ضبط مواضع المكونات على شبكة النقاط الموجودة فى الفورمة أو ضبط كل مكون بالنسبة للمكونات الأخرى.

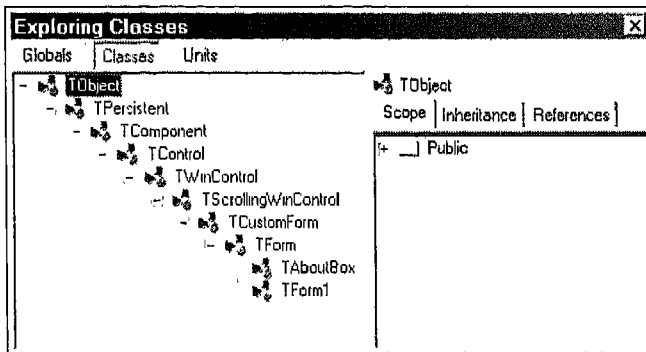


تستطيع أيضا ضبط مواضع المكونات عن طريق صندوق الحوار Alignment الموضح فى الشكل التالى :

الأمر Browser بالقائمة View

اختيار الأمر Browser من القائمة View يؤدي إلى فتح النافذة Project Browser

الموضحة فى الشكل التالى :



شكل توضيحي :

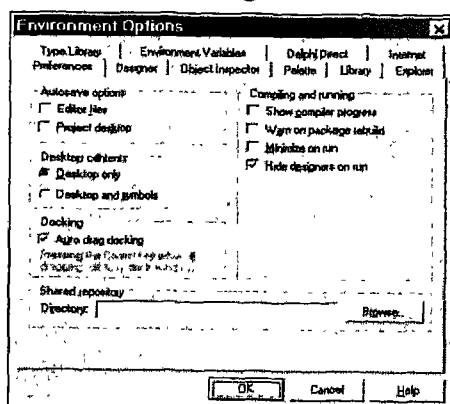
هناك طريقة أخرى لفتح النافذة Project Browser وهى الضغط على مجموعة المفاتيح Shift+Ctrl+B بلوحة المفاتيح.





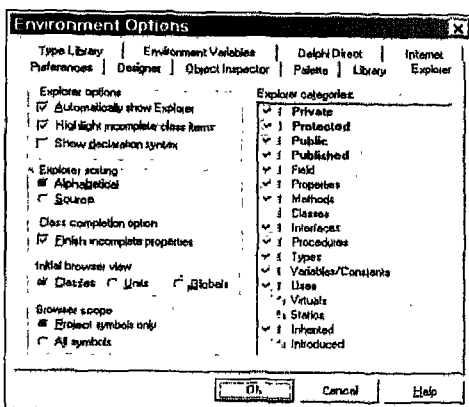
هذا ولكى تتمكن من تحديد مدى أو مجال عمل النافذة Project Browser اتبع الخطوات التالية :

(١) افتح القائمة Tools ثم اختر منها الأمر Environment Options ليظهر على الشاشة صندوق الحوار Environment Options الموضح فى الشكل التالى :



شكل توضيحي :

(٢) انقر بالماوس على التبويب Explorer ليظهر على السطح كما هو موضح بالشكل التالى :

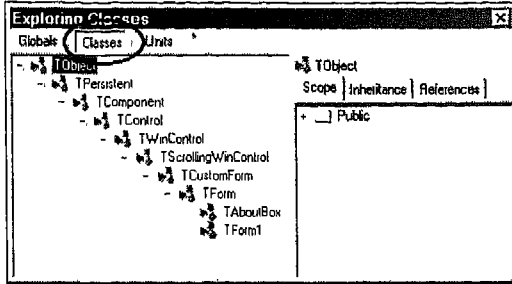


شكل توضيحي :

(٣) بهذا التبويب وفى القسم Browser Scope علم بالماوس إما على الاختيار Project symbols only أو على الاختيار All symbols الذى يتضمن المكتبة VCL ومن ثم تستطيع إما أن تشاهد الرموز من المشروع الذى تعده فقط أو من كافة الوحدات التى يستخدمها المشروع الذى تعده.

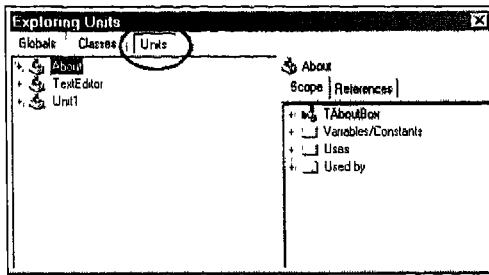


(٤) من خلال القسم Initial Browser view تستطيع أن تحدد المشهد الابتدائي للنافذة Project Browser بحيث يكون إما مشهد القطاعات classes الموضح في الشكل التالي :



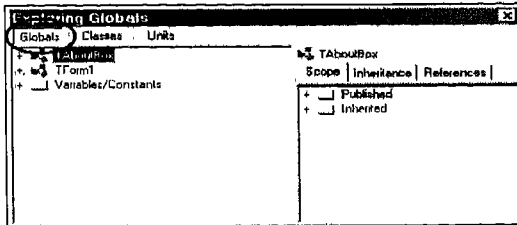
شكل توضيحي :

أو مشهد الوحدات Units الموضح في الشكل التالي :



شكل توضيحي :

أو مشهد العموميات globals الموضح في الشكل التالي :



شكل توضيحي :

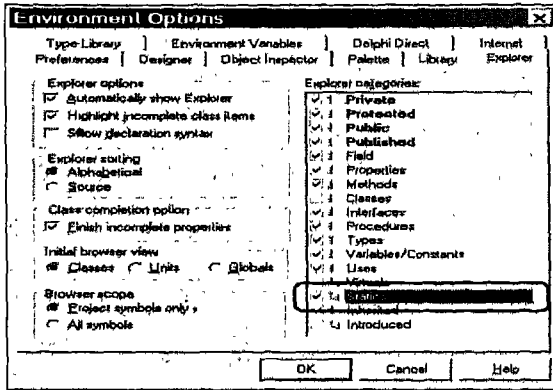
معنى ذلك أنه عندما تقوم بفتح النافذة Project Browser يظهر على الفور المشهد الابتدائي بهذه النافذة.



الخيارات الموجودة في القسم Explorer categories بالتبويب Explorer بصندوق الحوار Environment Options تسمح لك بأن تتحكم في الطريقة التي يتم بها



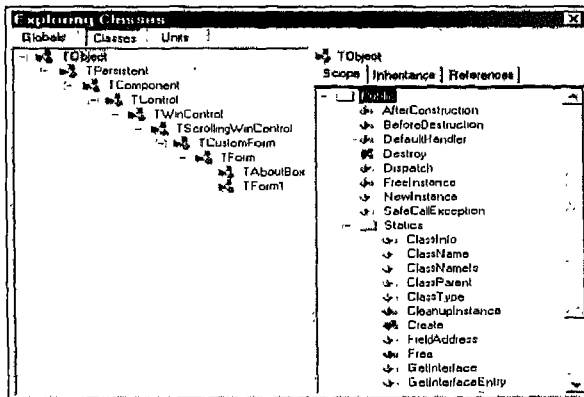
تجميع العناصر التي يتم عرضها داخل النافذة Project Browser. وفي حالة التعليم على أى اختيار من هذه الإختيارات تجد أنه ظهر عمود مناظر لهذا الاختيار فى النافذة Project Browser. فعلى سبيل المثال لو قمنا بالتعليم على الاختيار Statics كما هو موضح بالشكل التالى :



شكل توضيحي :

فى هذه الحالة تصبح العناصر المعروضة فى النافذة Project Browser كما هو

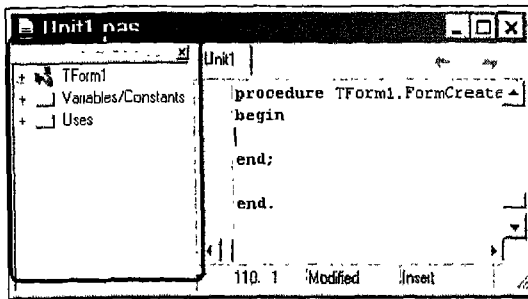
موضح بالشكل التالى :



شكل توضيحي :

الأمور Code Explorer بالقائمة View

من الطبيعى أن تكون النافذة Code Explorer راسية أو ملتصقة بيسار محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالى :

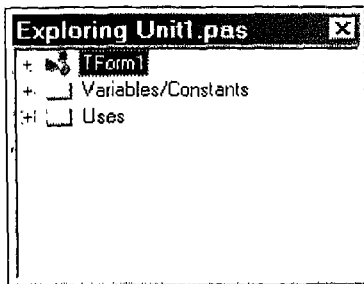


شكل توضيحي :

أما لو كانت هذه النافذة مغلقة تستطيع حينئذ إعادة فتحها عن طريق اختيار الأمر Code Explorer من القائمة View.

النافذة Code Explorer تجعل من السهولة بمكان التجول عبر ملفات الوحدات التي تتعامل معها بالإضافة إلى إمكانية إنشاء القطاعات تلقائياً.

● لكي تغلق النافذة Code Explorer في هذه الحالة ينبغي جعلها منفصلة عن نافذة محرر الكود البرمجي Code Editor لتصبح كما هو موضح بالشكل التالي :

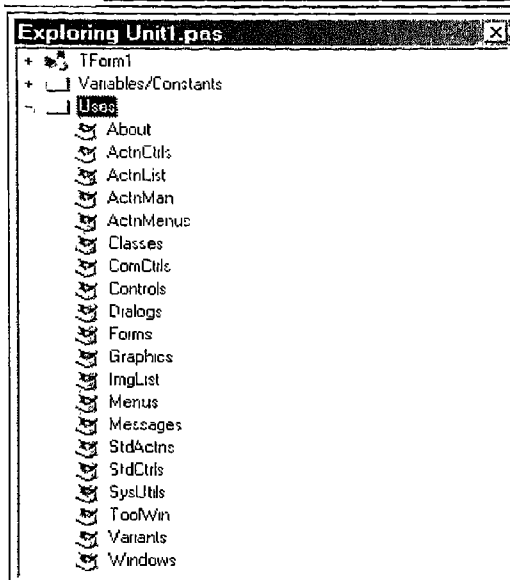


شكل توضيحي :

ثم تنقر بالماوس على الزر (X) بالركن الأيمن العلوي للنافذة.

● لكي تعيد فتح النافذة Code Explorer اختر الأمر Code Explorer من القائمة View.

النافذة Code Explorer تشتمل على ديجرام شجري يوضح كافة أنواع القطاعات والخصائص والأساليب والمتغيرات العامة والروتينات العامة التي تم تعريفها في ملف الوحدة الذي تتعامل معه. كما إنه يوضح أيضاً الوحدات الأخرى من خلال المجلد uses كما هو موضح بالشكل التالي :



شكل توضيحي :

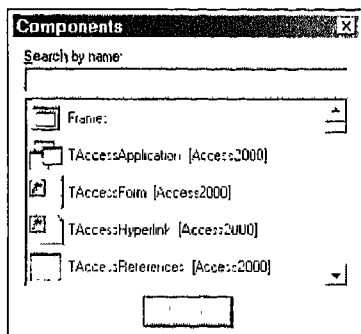
تستطيع فتح أو غلق المجلدات الموجودة فى المشهد الشجرى وذلك عن طريق العلامة (+) أو (-) الموجودة بجوار اسم المجلد الذى ترغب فى التعامل معه.

أى ملف وحدة يكون مفتوح فى نافذة محرر الكود البرمجى Code Editor فإنه يكون مفتوح أيضا فى النافذة Code Explorer.



الأمر Component List بالقائمة View

اختيار الأمر Component List من القائمة View يؤدي إلى عرض نافذة المكونات Components على الشاشة كما هو موضح بالشكل التالى :



شكل توضيحي :

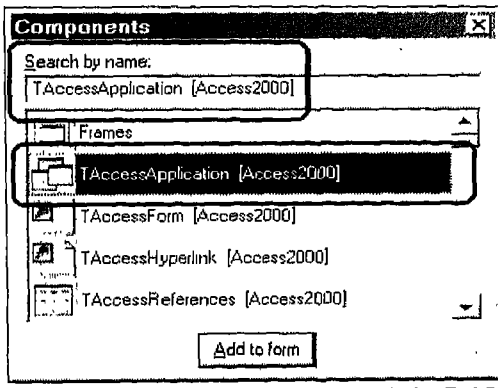


نافذة المكونات Components

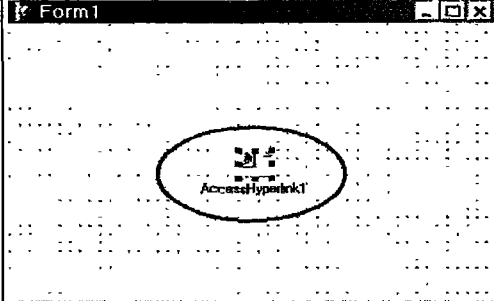
يمكن استخدام نافذة المكونات Components لإضافة مكونات إلى الفورم التى تعدها بمشروعك وذلك باستخدام الماوس أو من خلال لوحة المفاتيح.

هذا والجدول التالى يقدم لنا وصف مختصر للعناصر التى تتألف منها نافذة المكونات Components :

العنصر	الوصف والاستخدام
الحقل Search by name	فى هذا الحقل يتم ادخال اسم المكون الذى ترغب فى إضافته للفورمة. وهذا الحقل يؤدي مهمة بحث تزايدية ومن ثم تجد أن المؤشر يتحرك إلى المكون الذى يكون اسمه مشتملا على الحروف التى تكتبها فى أثناء كتابتها بهذا الحقل.
قائمة العرض Component	عن طريق هذه قائمة العرض هذه يمكن أن تختار المكون الذى ترغب فى إضافته. هذا وستجد أن المكونات معروضة فى هذه القائمة وهى مرتبة ترتيب أبجدى كما ستجد أن كل مكون يوجد بجواره الأيقونة التى تمثله.
الحقل Search	عندما تختار مكون فإن اسمه يظهر فى الحقل Search كما هو موضح بالشكل التالى :





<p>النقر بالماوس على المفتاح Add To Form يؤدي إلى وضع حالة instance للمكون المختار وستلاحظ أن المكون قد تم وضعه في مركز الفورمة تماما كما هو موضح بالشكل التالى :</p>	<p>المفتاح Add to form</p>
	

لكى تختار Add To Form من لوحة المفاتيح اضغط على المفتاح Enter بلوحة المفاتيح.



لكى تضيف المكون الذى اخترته فى صندوق الحوار Component List عليك أن تقوم بواحد من الآتى :

- الضغط على المفتاح Enter بلوحة المفاتيح.
- النقر بالماوس نقرا مزدوجا على اسم المكون فى قائمة العرض Components.
- النقر بالماوس على اسم المكون فى قائمة العرض Components ثم النقر بالماوس على المفتاح Add To Form.

عندما تضيف مكون إلى فورمة عن طريق استخدام لوحة المفاتيح تجد أن لغة Delphi تستخدم الحجم الافتراضى للمكون كما إنها تضيف المكون فى مركز الفورمة إذا لم يكن هناك مكون حاوية (مثل صندوق التجميع أو اللوحة panel) مختارا فى الفورمة.

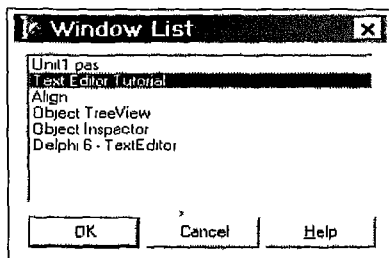


فى حالة وجود مكون حاوية مختار فى الفورمة فى هذه الحالة تقوم اللغة بوضع المكون الذى تضيفه فى مركز هذا المكون الحاوية. على العموم لكى تضيف مكون داخل حاوية فإنه ينبغى عليك اختيار الحاوية أولا قبل النقر بالماوس على المفتاح Add to form فى صندوق الحوار Component List.



الأمر Window List (بالقائمة View أو القائمة Window)

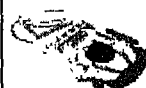
اختيار الأمر Window List من القائمة View أو من القائمة Window يؤدي إلى فتح صندوق الحوار Window List يضم قائمة بكافة النوافذ المفتوحة حاليا فى بيئة التطوير المتكاملة IDE. وهذا الشكل يوضح لنا صندوق الحوار هذا :



شكل توضيحي :

عند اختيار اسم نافذة من النوافذ المعروضة فى صندوق الحوار Window List فإن ذلك يؤدي إلى جعل هذه النافذة نشطة أى يتم نقل دفة التحكم لها. ولو أن لديك الكثير من النوافذ مفتوحة فى هذه الحالة ستجد أن صندوق الحوار Window List يعتبر أسهل طريقة للذهاب إلى النافذة التى ترغبها.

هناك طريقة أخرى لفتح صندوق الحوار Window List وهى الضغط على المفاتيح Alt+O بلوحة المفاتيح.



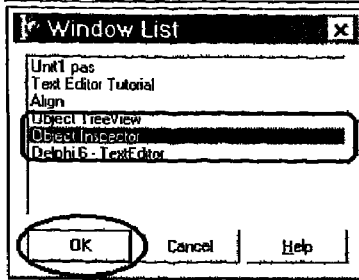
صندوق الحوار Window List

يمكن استخدام صندوق الحوار Window List لتحويل نافذة من كونها غير نشطة لتصبح نشطة. هذا ويقوم صندوق الحوار Window List باستعراض كافة النوافذ المفتوحة حاليا فى المشروع.

لكى تختار نافذة قم بأى من الآتى :

● النقر بالماوس نقرا مزدوجا على اسم النافذة.

● اختيار اسم النافذة ثم النقر بالماوس على المفتاح Ok كما هو موضح بالشكل التالى :



شكل توضيحي :

الأمر Debug Windows بالقائمة View

يمكن استخدام مجموعة الأوامر الموجودة بالقائمة الفرعية الخاصة بالأمر Debug Windows بالقائمة View لفتح النوافذ التى تنتمى إلى الأداة المتكاملة لاكتشاف الأخطاء ومعالجتها Integrated debugger. وهذه النوافذ توضح لنا كيف تتم عملية اكتشاف الثغرات التى توجد بالبرنامج ومعالجتها أيضا.

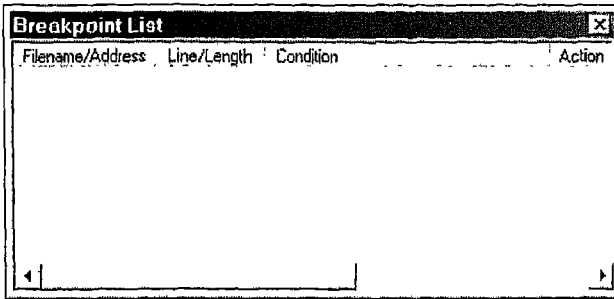
الجدول التالى يقدم لنا وصف مختصر للأوامر الموجودة بالقائمة الفرعية الخاصة بالأمر Debug Windows بالقائمة View :

الوصف والاستخدام	الأمر
يعمل هذا الأمر على عرض نافذة المراقبة Breakpoint List.	Breakpoints
من خلال هذا الأمر يتم فتح نافذة المراقبة Call Stack.	Call Stack
عن طريق هذا الأمر يتم فتح نافذة المراقبة Watch List.	Watches
يعمل هذا الأمر على توضيح المتغيرات المحلية الخاصة بالدالة التى يتم تنفيذها حاليا فى أثناء كون البرنامج فى مود الـ debug.	Local Variables
من خلال هذا الأمر يتم فتح نافذة المراقبة Thread Status.	Threads
من خلال هذا الأمر يتم فتح نافذة المراقبة Modules.	Modules
يعمل هذا الأمر يتم فتح نافذة المراقبة Event Log.	Event Log
يعمل هذا الأمر يتم فتح نافذة المراقبة CPU.	CPU
يعمل هذا الأمر يتم فتح نافذة المراقبة FPU.	FPU



الأمر Breakpoints بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Breakpoints من القائمة الفرعية Debug Windows من القائمة View يؤدي إلى فتح النافذة Breakpoint List الموضحة فى الشكل التالى :



شكل توضيحي :

هناك طريقة أخرى لفتح النافذة Breakpoint List وهى الضغط على مجموعة المفاتيح Ctrl+Alt+B بلوحة المفاتيح.



لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى النافذة Breakpoint List.



النافذة Breakpoint List تقوم بعرض كافة نقط التوقف المحددة حالياً. وكل نقطة توقف موجودة فى هذه النافذة نشاهد لها المعلومات الموضحة فى الجدول التالى :

العمود	الوصف والاستخدام
Filename/Address	هذا العمود يعرض اسم الملف الذى تم فيه تحديد نقطة التوقف. كما يعمل أيضا على عرض اسم المتغير الذى تم فيه تحديد البيانات الخاصة بنقط التوقف. كذلك يمكن أن يعرض هذا العمود العناوين التى تم عندها تحديد نقط التوقف وذلك إذا لم يكن العنوان مرتبط بالسطر الذى تم فيه تحديد نقط التوقف وفى هذه الحالة يتم عرض اسم الملف الموجود به هذا السطر.
Line/Length	بالنسبة لـ Source breakpoints نجد أن هذا العمود يعرض رقم

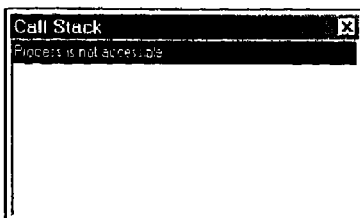


السطر الموجود فيه نقطة التوقف. أما بالنسبة لـ Data breakpoints فإن هذا العمود يعرض طول نقطة التوقف. وبالنسبة لـ Address breakpoints يكون هذا العمود فارغ وذلك إذا لم تكن هناك إمكانية لربط العنوان بالسطر الموجود فيه لـ Source breakpoints ولكن فى حالة تحقق هذا الارتباط فسنجد أن هذا العمود يعرض رقم هذا السطر بالإضافة إلى الكود السداسى عشر للعنوان من بداية السطر.	
هذا العمود يعرض أى حالة تكون مرتبطة بنقطة التوقف التى يجرى عرضها الآن.	Condition
هذا العمود يعرض أى رقم مرور مرتبط بنقطة التوقف التى يجرى عرضها الآن.	Pass Count

الأمر Call Stack بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Call Stack من القائمة الفرعية Debug Windows من القائمة

View يؤدى إلى فتح النافذة Call Stack الموضحة فى الشكل التالى :



شكل توضيحى :

هناك طريقة أخرى لفتح النافذة Call Stack وهى الضغط على مجموعة المفاتيح Ctrl+Alt+S بلوحة المفاتيح.



لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى النافذة Call Stack.

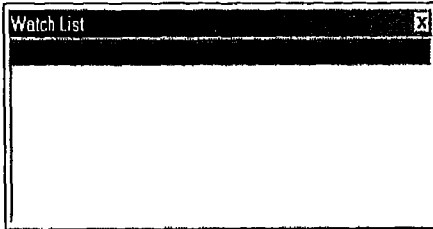




النافذة Call Stack تعرض التتابع الحالى للروتينات التى قام البرنامج باستدعاؤها فى مرحلة التشغيل Run-Time. وفى هذا العرض نجد أن آخر روتين تم استدعاؤه موجود فى القمة القائمة. هذا وكل عنصر فى النافذة Call Stack يعرض معلومات متمثلة فى كل من اسم الإجراء والقيم الخاصة بأى معاملات يتم تمريرها إليه.

الأمر Watches بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Watches من القائمة الفرعية Debug Windows من القائمة View يؤدي إلى فتح نافذة المراقبة Watch List الموضحة فى الشكل التالى :



شكل توضيحي :

هناك طريقة أخرى لفتح نافذة المراقبة Watch List وهى الضغط على مجموعة المفاتيح Ctrl+Alt+W بلوحة المفاتيح.



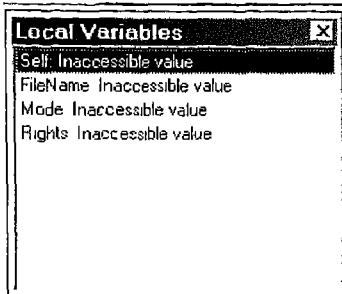
لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى نافذة المراقبة Watch List.



نافذة المراقبة Watch List تعرض كافة تعبيرات المراقبة التى تم تحديدها حالياً. ولو أنك احتفظت بهذه النافذة مفتوحة فى أثناء كون البرنامج فى مود الـ Debug (مود اكتشاف الثغرات البرمجية ومعالجتها) فى هذه الحالة تستطيع أن تراقب سلوك البرنامج وهو يقوم بتحديث القيم المخصصة للمتغيرات الهامة فى أثناء تشغيل البرنامج.

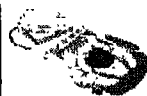
الأمر Local Variables بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Local Variables من القائمة الفرعية Debug Windows من القائمة View يؤدي إلى عرض القيم المخصصة حالياً للمتغيرات المحلية الخاصة بالدالة التى يتمك التعامل معها حالياً وذلك فى أثناء كون البرنامج فى مود الـ debug وذلك من خلال نافذة المراقبة Local Variables الموضحة فى الشكل التالى :



شكل توضيحى :

هناك طريقة أخرى لفتح نافذة المراقبة Local Variables وهى الضغط على مجموعة المفاتيح Ctrl+Alt+L بلوحة المفاتيح.



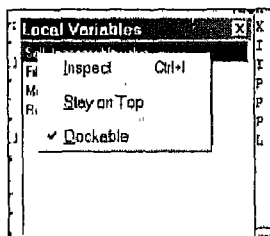
لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى نافذة المراقبة Local Variables.



هذا الأمر يكون دائما متاح ولكن نافذة المراقبة Local Variables تكون فارغة إذا لم تكن أداة فحص الأخطاء ومعالجتها debugger معلقة paused. هذا ولو أنك احتفظت بهذه النافذة مفتوحة فى أثناء القيام بمهام اكتشاف الأخطاء ومعالجتها فى هذه الحالة تستطيع تراقب سلوك البرنامج وهو يقوم بتحديث القيم المخصصة للمتغيرات الهامة فى أثناء تشغيل البرنامج..

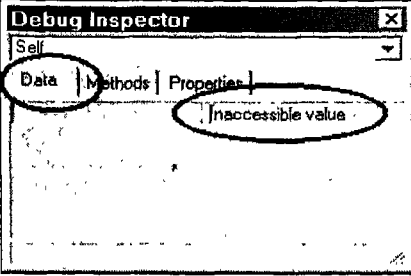
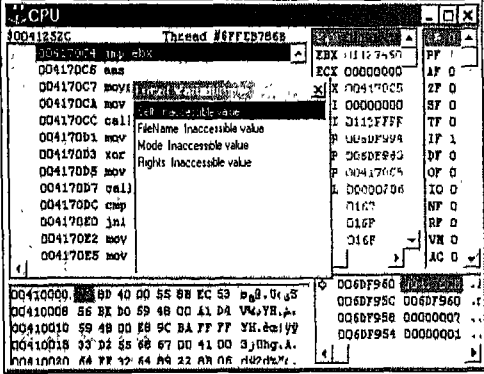
القائمة المختصرة الخاصة بنافذة المراقبة Local Variables

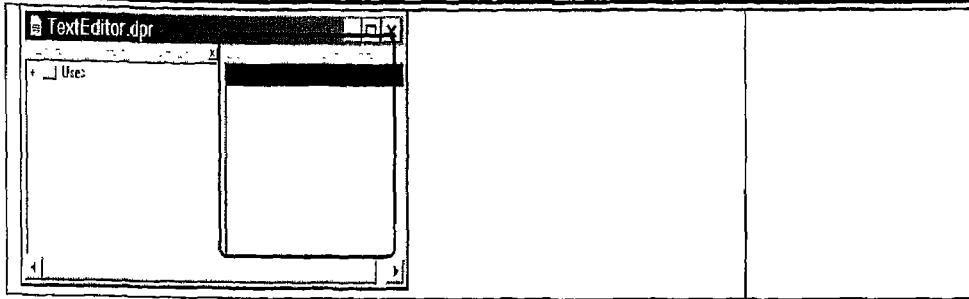
عند النقر بالزر الأيمن للماوس على أى عنصر من العناصر الموجودة فى نافذة المراقبة Local Variables تظهر على الشاشة القائمة المختصرة الموضحة فى الشكل التالى :



شكل توضيحى :

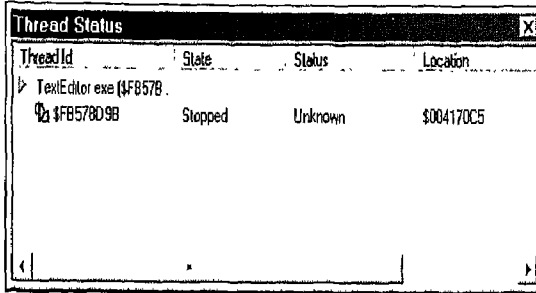
الجدول التالى يقدم لنا وصف مختصر لوظيفة الأوامر الموجودة فى القائمة المختصرة الموضحة فى الشكل السابق.

الوصف والاستخدام	الأمر
<p>يقوم هذا الأمر بعرض معلومات عن المتغير المختار حاليا وهذه المعلومات يتم عرضها فى التوبيب Data بالنافذة Debug Inspector كما هو موضح بالشكل التالى :</p>  <p>هذا ويمكن تنفيذ الأمر Inspect عن طريق الضغط على المفاتيح Ctrl+I بلوحة المفاتيح.</p>	<p>Inspect</p>
<p>يعمل هذا الأمر على الاحتفاظ بالنافذة Local Variables ظاهرة على الشاشة حتى ولو لم تكن نشطة كما هو موضح بالشكل التالى :</p> 	<p>Stay On Top</p>
<p>هذا الأمر يسمح للنافذة Local Variables بأن يكون لديها القدرة على الإرساء أو الإلتصاق بأى نافذة من النوافذ الأخرى الموجودة فى بيئة التطوير المتكاملة IDE كما هو موضح بالشكل التالى الذى يوضح لنا النافذة Local Variables وهى ملتصقة بنافذة محرر الكود البرمجى Code Editor :</p>	<p>Dockable</p>

**الأمر Threads بالقائمة الفرعية Debug Windows من القائمة View**

اختيار الأمر Threads من القائمة الفرعية Debug Windows من القائمة View

يؤدى إلى فتح نافذة المراقبة Thread Status الموضحة فى الشكل التالى :



شكل توضيحي :

هناك طريقة أخرى لفتح نافذة المراقبة Thread Status وهى الضغط على مجموعة المفاتيح Ctrl+Alt+T بلوحة المفاتيح.



لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى فتح نافذة المراقبة Thread Status.



نافذة المراقبة Thread Status تعرض حالة كافة threads التى يتم تنفيذها حالياً فى التطبيق الذى يكون فى مود Debug حالياً. وفى هذا الصدد نقول إن عملية الـ debugging للعديد من العمليات التى تتم فى نفس الوقت فيمكن تدعيمها أيضاً وذلك باستخدام نافذة المراقبة Thread Status.



الأمر Modules بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Modules من القائمة الفرعية Debug Windows من القائمة View يؤدي إلى فتح نافذة المراقبة Modules الموضحة في الشكل التالي :

Name	Type	Address	Path	Entry Point	Address
Process: JPBST0023					
TestData.exe	10-400000		C:\Program Files\Borland\		
winlibm.dll	18010000		C:\WINDOWS\SYSTEM		
SHELL32.dll	19720000		C:\WINDOWS\SYSTEM		
MSVB60.dll	17400000		C:\WINDOWS\SYSTEM		
gdi32.dll	47020000		C:\WINDOWS\SYSTEM		
comctl32.dll	17610000		C:\WINDOWS\SYSTEM		
USER32.dll	18720000		C:\WINDOWS\SYSTEM		
USER32.dll	18720000		C:\WINDOWS\SYSTEM		
ole32.dll	85600000		C:\WINDOWS\SYSTEM		
OLEAUT32.dll	49740000		C:\WINDOWS\SYSTEM		
ADVAPI32.dll	103E0000		C:\WINDOWS\SYSTEM		

شكل توضيحي :

هناك طريقة أخرى لفتح نافذة المراقبة Modules وهي الضغط على مجموعة المفاتيح **Ctrl+Alt+M** بلوحة المفاتيح.



لا تحتاج لأن يكون البرنامج في الـ debug عندما ترغب في فتح نافذة المراقبة Modules.

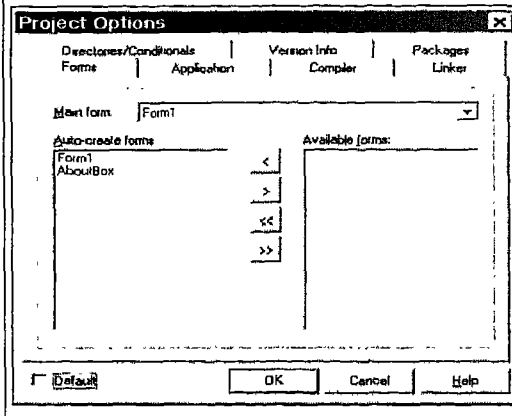


من خلال نافذة المراقبة Modules يمكن أن نشاهد قائمة بالـ Modules التي تم تحميلها بالذاكرة عند تشغيل المشروع الحالي. وفي هذا الصدد نقول إن الـ Module يمكن أن يكون ملف تنفيذي أو ملف مكتبة الربط الديناميكي DLL أو حزمة برمجية يحتاج المشروع الحالي أن يتم تحميلها بالذاكرة في أثناء مرحلة التشغيل Run-Time. ونود هنا القول بأن نافذة المراقبة Modules تعرض اسم كل module وصورته في مرحلة التشغيل Run-Time وعنوان الأساس الخاص بها بالإضافة إلى المسار الدال على موضع تحميل الـ module بالذاكرة.

من المعتاد أن يتم فتح هذه النافذة بعد أن تنتهي من ترجمة المشروع وبعد أن يتم فحصه وعلاج ما به من أخطاء وثرغات. ولكن من المفيد أيضا فتح هذه النافذة عند إجراء عملية optimizing للتقليل بقدر الإمكان من وقت التحميل وذلك عن طريق تحديد الترحيلات الأساسية للصورة المفضلة لكل Module مطلوب.

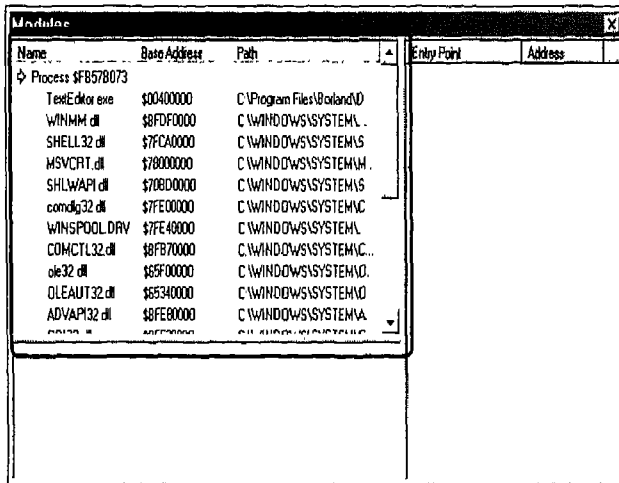


عنوان الأساس لصورة التشغيل للـ Module عبارة عن offset لذاكرة الكمبيوتر ويكون بالكود السداسى عشر وهو الموضع الذى يتم فيه بشكل حقيقى تحميل الـ Module كشيء مميز من عنوان الأساس للصورة المفضلة والذى يمكن تحديده من خلال صندوق الحوار Project Options الموضح فى الشكل التالى :



نافذة المراقبة Modules تتألف من الأجزاء الثلاثة التالية :

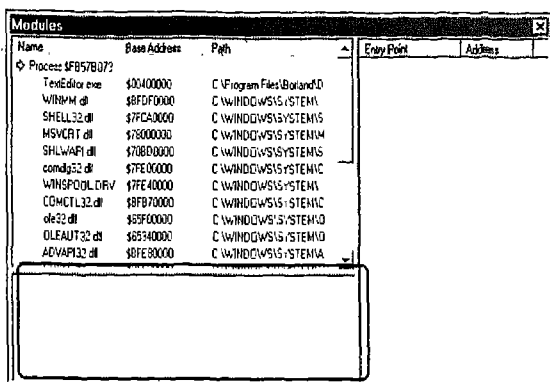
قسم الـ Module الموجودة فى الركن الأيسر العلوى بنافذة المراقبة Modules كما هو موضح بالشكل التالى :



شكل توضيحى :

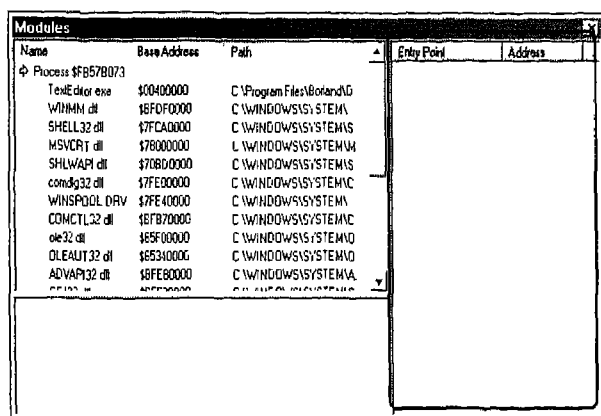


قسم المصدر Source pane بالركن الأيسر السفلي بنافذة المراقبة Modules كما هو موضح بالشكل التالي :



شكل توضيحي :

قسم نقطة الإدخال Entry point pane بالجانب الأيمن لنافذة المراقبة Modules كما هو موضح بالشكل التالي :



شكل توضيحي :

القسم Module يعرض اسم كل module وعنوانه بالموضع الذي يتم تحميله فيه بذاكرة الكمبيوتر. ولو أن module لديه معلومات عن تصحيح ومعالجة الأخطاء (debug) في هذه الحالة نجد أن القسم Source pane يعرض مشهد شجري للملفات المصدرية أو الأصلية التي تشتمل على الكود البرمجي الذي تم استخدامه لبناء module كما أن القسم Entry point يعرض قائمة بكافة الرموز العامة كما هو موضح بالشكل التالي :

[illegible]

شکل توضیحی :

أما لو كان الـ Module ليس لديه معلومات debug في هذه الحالة نجد أن القسم Entry point يعرض نقط إدخال الدوال بالـ Module كما هو موضح بالشكل التالي :

Name	Base Address	Path	Entry Point	Address
OLEAUT32.dll	105A4000	C:\WINDOWS\SYSTEM32\OLEAUT32.dll	AbscXProc	18FFD2C03
ADVAPI32.dll	1BFA0000	C:\WINDOWS\SYSTEM32\ADVAPI32.dll	AdvApi32	18FFD2C03
ADSI-1.dll	1BF6F000	C:\WINDOWS\SYSTEM32\ADSI-1.dll	AddFunctionUseAd	18FFD2B49
ADSI-2.dll	1BF70000	C:\WINDOWS\SYSTEM32\ADSI-2.dll	AddFunctionUseAdW	18FFD27A5
USER32.dll	1BF85000	C:\WINDOWS\SYSTEM32\USER32.dll	Analogous	18FFD2742
VERRIES.LIB	1BF70000	C:\WINDOWS\SYSTEM32\VERRIES.LIB	AnimalPalette	18FFD2E61
VERIFIER.DLL	1BF70000	C:\WINDOWS\SYSTEM32\VERIFIER.DLL	Arc	18FFD222C
IMM32.dll	1BF82000	C:\WINDOWS\SYSTEM32\IMM32.dll	AsTo	18FFD2A54
ShellExec.dll	15C00000	C:\WINDOWS\SYSTEM32\ShellExec.dll	BegunPath	18FFD2C67
MVCRT10.dll	170A0000	C:\WINDOWS\SYSTEM32\MVCRT10.dll	DMA	18FFD2C93
MVCRT40.dll	17750000	C:\WINDOWS\SYSTEM32\MVCRT40.dll	DynLinkDll	18FFD2A7A
RICHED20.dll	17F6C000	C:\WINDOWS\SYSTEM32\RICHED20.dll	GCMC	18FFD2A15
HPCRT10.dll	17010000	C:\WINDOWS\SYSTEM32\HPCRT10.dll	IHookColorIndexAnd	18FFD2145
			CheckSpecOfControl	18FFD2703
			Close	18FFD2C3E
			CloseRhhDataFile	18FFD3A00
			CloseFigure	18FFD2CB8
			CloseMetafile	18FFD2AC0
			CreateCurrentPalette	18FFD2A4C
			ColorMatchToTarget	18FFD2CFF
			EndPage	18FFD2A9F

شکل توضیحی :

إجراء عملية الـ debugging لأكثر من عملية في نفس الوقت

بالنسبة لإمكانية إجراء عملية debugging لأكثر من عملية في نفس الوقت يتم عرض كل عملية وكافة Modules المرتبطة بها وفي أثناء ذلك ستلاحظ أن العملية التي تجري حاليا يشار إليها بسهم أخضر كما هو موضح بالشكل التالي :

Name	Base Address	Path
Process: \\FB578073		
TextEditor.exe	00400000	C:\Program Files\Borland\D...
WINMM.dll	8BFD0000	C:\WINDOWS\SYSTEM\...
SHELL32.dll	7FCA0000	C:\WINDOWS\SYSTEMS...
MSVCRT.dll	78000000	C:\WINDOWS\SYSTEMM...
SHLWAPI.dll	70B00000	C:\WINDOWS\SYSTEMS...
comdlg32.dll	7FE00000	C:\WINDOWS\SYSTEMC...
WINSPOOL.DRV	7FE40000	C:\WINDOWS\SYSTEM\...
COMCTL32.dll	8BF70000	C:\WINDOWS\SYSTEMC...
ole32.dll	65F00000	C:\WINDOWS\SYSTEMO...
OLEAUT32.dll	65340000	C:\WINDOWS\SYSTEMO...
ADVAPI32.dll	8BF80000	C:\WINDOWS\SYSTEMA...
GDI32.dll	4D5E0000	C:\WINDOWS\SYSTEMC...

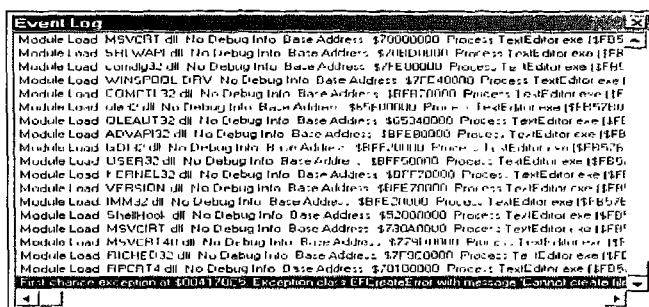
شكل توضيحي :



الأمر Event Log القائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Event Log من القائمة الفرعية Debug Windows من القائمة

View يؤدي إلى فتح نافذة المراقبة Event Log الموضحة فى الشكل التالى :



شكل توضيحي :

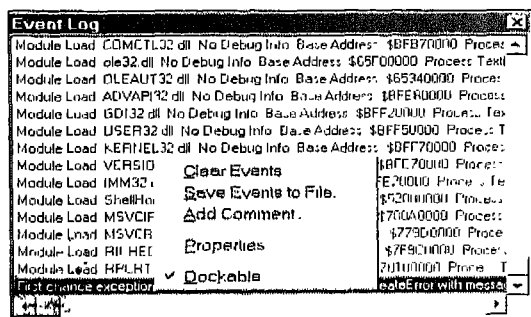
هناك طريقة أخرى لفتح نافذة المراقبة Event Log وهى الضغط على مجموعة المفاتيح Ctrl+Alt+V بلوحة المفاتيح.



لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى فتح نافذة المراقبة Event Log.



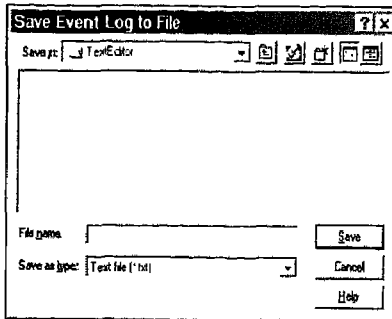
نافذة المراقبة Event Log تقدم لنا كل من الرسائل الخاصة بعملية التحكم والرسائل الخاصة بنقط التوقف ورسائل الويندوز. ونود هنا القول بأن النقر بالزر الأيمن للماوس على أى عنصر من العناصر الموجودة فى نافذة المراقبة Event Log يؤدي إلى عرض القائمة المختصرة الموضحة فى الشكل التالى :



شكل توضيحي :

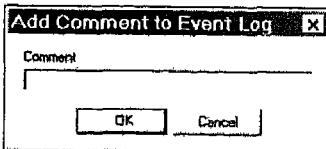


هذه القائمة المختصرة تشتمل على مجموعة من الأوامر التي تسمح لك بأن تنظف نافذة المراقبة Event Log من محتوياتها وذلك من خلال الأمر Clear Events كما تسمح لك بأن تحفظ محتويات هذه النافذة في ملف نصي وذلك من خلال الأمر Save Events to File والذي يؤدي إلى ظهور صندوق الحوار Save Events to File الموضح في الشكل التالي :



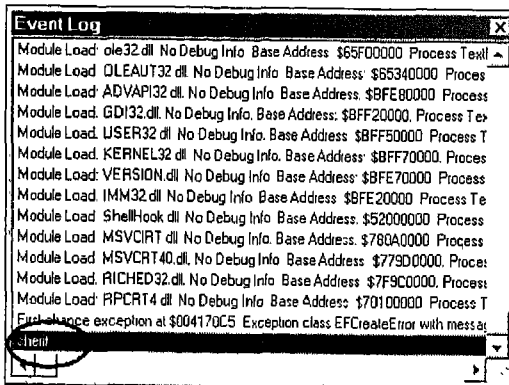
شكل توضيحي :

أما الأمر Add Comment فيعمل على إضافة تعليق إلى نافذة المراقبة Event Log وذلك من خلال صندوق الحوار Add Comment to Event Log الموضح في الشكل التالي :



شكل توضيحي :

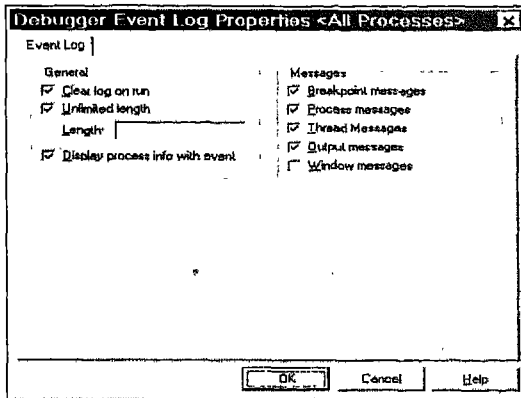
وستلاحظ أن التعليق الذي أضفته يظهر في آخر النافذة كما هو موضح بالشكل التالي :



شكل توضيحي :



أما الأمر Properties بالقائمة المختصرة السالفة الذكر فيعمل على تحديد قيم الخصائص الخاصة بنافذة المراقبة Event Log وذلك من خلال صندوق الحوار الموضح في الشكل التالي :

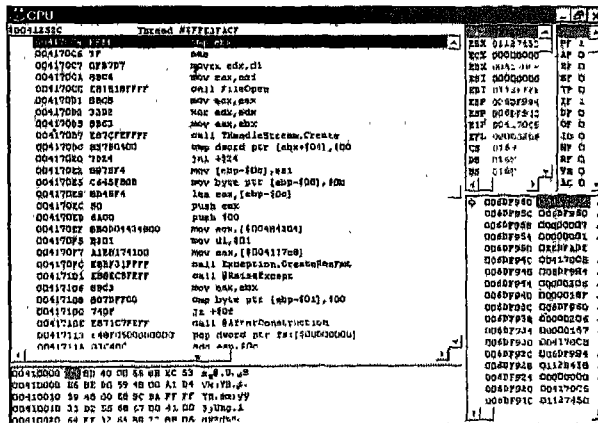


شكل توضيحي :

عن طريق تحديد الخصائص تستطيع أن تتحكم في عدد الرسائل التي يتم عرضها في نافذة المراقبة Event Log كما تستطيع أيضا التحكم في نوعية الأحداث التي يتم إظهارها بهذه النافذة.

الأمر CPU بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر CPU من القائمة الفرعية Debug Windows من القائمة View يؤدي إلى فتح النافذة CPU الموضحة في الشكل التالي :



شكل توضيحي :



هناك طريقة أخرى لفتح النافذة CPU وهى الضغط على مجموعة المفاتيح Ctrl+Alt+C بلوحة المفاتيح.



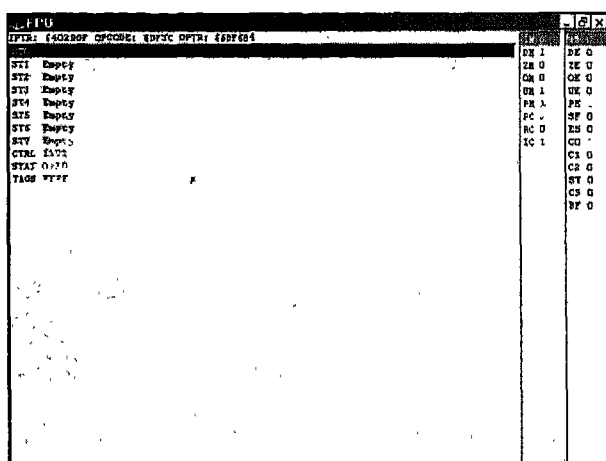
ستحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى فتح النافذة CPU.



من خلال النافذة CPU يتم إجراء العملية debugging لمستوى معين فى التطبيق مثل محتويات حزمة البرنامج أو registers أو الإشارات CPU flags أو العناصر التى ليس لها لزوم فى الذاكرة أو تعليمات التجميع المبعثرة من ماكينة الكود البرمجى الخاص بالتطبيق.

الأمر FPU بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر FPU من القائمة الفرعية Debug Windows من القائمة View يؤدي إلى فتح النافذة CPU الموضحة فى الشكل التالى :



شكل توضيحي :

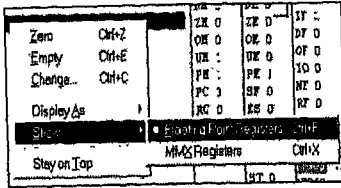
هناك طريقة أخرى لفتح النافذة FPU وهى الضغط على مجموعة المفاتيح Ctrl+Alt+F بلوحة المفاتيح.



ستحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى فتح النافذة FPU.

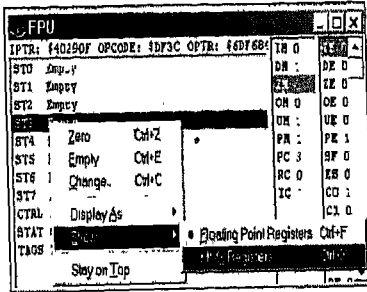


تستطيع أن تستخدم النافذة FPU لمشاهدة محتويات المكون FPU الخاص بوحدة المعالجة المركزية CPU. هذا وتستطيع أن تجعل النافذة FPU تعرض معلومات من النوع floating-point وذلك عن طريق الضغط على المفاتيح Ctrl+F بلوحة المفاتيح أو عن طريق النقر بالزر الأيمن للماوس على أى عنصر بالنافذة ثم من القائمة المختصرة التى تظهر على الشاشة اختر الأمر Show ثم الأمر Floating-Point registers كما هو موضح بالشكل التالى :



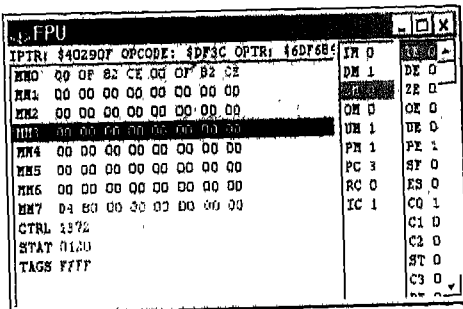
شكل توضيحي :

كما تستطيع أن تجعل النافذة FPU تعرض معلومات من النوع MMX Registers وذلك عن طريق الضغط على المفاتيح Ctrl+X بلوحة المفاتيح أو عن طريق النقر بالزر الأيمن للماوس على أى عنصر بالنافذة ثم من القائمة المختصرة التى تظهر على الشاشة اختر الأمر Show ثم الأمر MMX Registers كما هو موضح بالشكل التالى :



شكل توضيحي :

وفى هذه الحالة تصبح المعلومات المعروضة فى النافذة FPU كما هو موضح بالشكل التالى



شكل توضيحي :



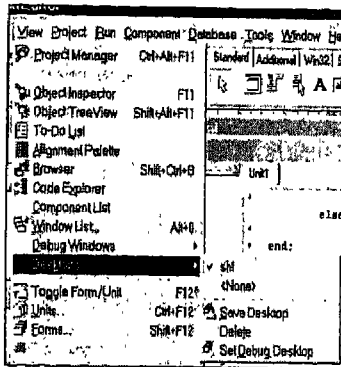
تقوم النافذة FPU بعرض كل من القيم والحالات الخاصة بكل Register فى FPU كما تعرض أيضا حالة الـ FPU وأدوات التحكم وكلمات الـ tags. وفى هذا الصدد نقول إن الـ flags يتم تكويدها فى أداة التحكم كما يتم عرض الكلمة Status فى أجزاء أخرى منفصلة فى النافذة. كذلك تستطيع أيضا مشاهدة كل من العنوان والـ opcode والـ operand التى تناظر لآخر تعليمة FPU تم تنفيذها.

الأمر Desktops بالقائمة View

من خلال الأمر Desktops الموجودة بالقائمة View نستطيع مشاهدة تخطيطات مختلفة لسطح المكتب الخاصة ببيئة التطوير المتكاملة IDE التى قمنا بحفظها قبل ذلك كما نستطيع أيضا من خلال هذا الأمر حفظ تخطيطات جديدة لسطح المكتب.

الشكل التالى يوضح لنا الأوامر الموجودة بالقائمة الفرعية الخاصة بالأمر Desktops

بالقائمة View :

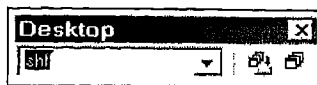


شكل توضيحي :

الجزء العلوى بهذه القائمة الفرعية (فوق الأمر Save Desktop) يعرض التخطيطات



التي سبق حفظها.

كذلك تستطيع استخدام شريط الأدوات Desktops الموضح فى الشكل التالى :



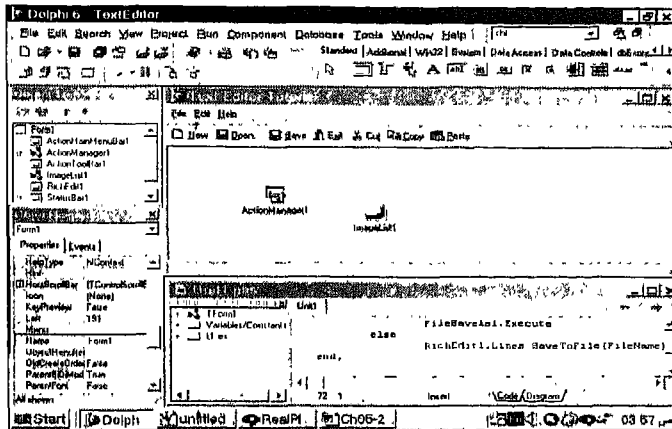
شكل توضيحي :



فمن خلال شريط الأدوات Desktops تستطيع أن تختار تخطيط سطح المكتب الذى ترغبه. فشريط الأدوات Desktops يشتمل على قائمة منسدلة تضم مجموعة من التخطيطات المختلفة لسطح المكتب كما تضم أيضا مجموعة من الأيقونات التى تسمح لك بأن تحفظ التخطيط الحالى لسطح المكتب  أو لجعل التخطيط الحالى لسطح المكتب عبارة عن سطح المكتب للعملية debugging .


الأمر Save Desktop بالقائمة الفرعية للأمر Desktops بالقائمة View

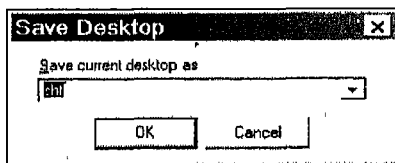
لكى تحفظ التخطيط الحالى لسطح المكتب اتبع الخطوات التالية :
(١) قم بترتيب عناصر سطح المكتب بالطريقة التى ترغبها وفى أثناء ذلك يمكن أن تنقل أى عنصر وتقوم بتغيير حجم أى عنصر وتجعل عناصر ملتصقة ببعضها البعض كما هو موضح بالشكل التالى (كمثال):



شكل توضيحي :

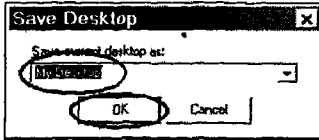
(٢) اختر الأمر Save Desktop من القائمة الفرعية للأمر Desktops بالقائمة View

أو انقر بالماوس على الأيقونة Save Current Desktop  بشريط الأدوات Desktops ليظهر على الشاشة صندوق الحوار Save Desktop الموضح فى الشكل التالى :



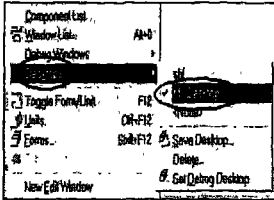
شكل توضيحي :

(٣) فى حقل القائمة المنسدلة Save current desktop as اكتب الاسم الذى ترغب فى تخصيصه لهذا التخطيط لعناصر سطح المكتب وليكن My Desktop كما هو موضح بالشكل التالى ثم انقر بالماوس على المفتاح Ok :



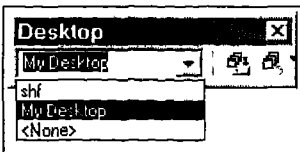
شكل توضيحي :

الآن عندما تفتح القائمة View وتختار منها الأمر Desktops ستشاهد العنصر My Desktop فى الجزء العلوى بالقائمة الفرعية الخاصة بهذا الامر كما هو موضح بالشكل التالى :



شكل توضيحي :

كما أن القائمة المنسدلة الموجودة فى شريط الأدوات Desktops أصبحت مشتملة على العنصر My Desktop كما هو موضح بالشكل التالى :



شكل توضيحي :

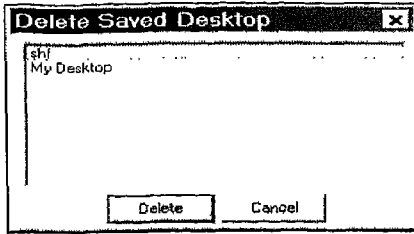
هذا التخطيط يتم حفظه لاستخدامه بعد ذلك مع كافة المشاريع التى سيتم استخدامها عندما تبدأ فى تشغيل لغة Delphi مرة أخرى. وبهذه الطريقة تستطيع أن تنشأ ما يحلو لك من تخطيطات وتستخدمها بعد ذلك مع المشاريع التى تتولى إعدادها.

الأمر Delete بالقائمة الفرعية للأمر Desktops بالقائمة View

يعمل الأمر Delete بالقائمة الفرعية للأمر Desktops بالقائمة View على إظهار النافذة Delete Saved Desktop الموضحة فى الشكل التالى :

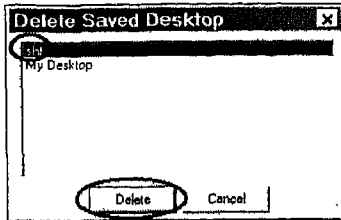


شكل توضيحي :



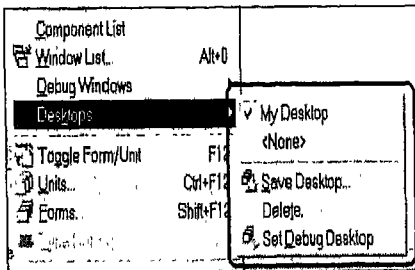
وهذه النافذة تعرض اسماء التخطيطات التى سبق حفظها ومن ثم تستطيع أن تسمح أى منها وذلك بأن تعلم على اسم التخطيط الذى ترغب فى مسحه وليكن مثلا التخطيط المسمى shf (الموضح فى الشكل التالى) ثم تنقر الماوس على المفتاح Delete كما هو موضح بالشكل التالى :

شكل توضيحي :



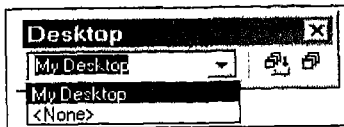
والآن عندما تفتح القائمة الفرعية للأمر Desktops فلن تجد بها التخطيط shf كما هو موضح بالشكل التالى :

شكل توضيحي :



كما لن تجده فى القائمة المنسدلة بشريط الأدوات Desktops كما هو موضح بالشكل التالى :

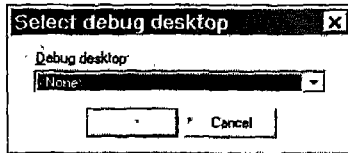
شكل توضيحي :



**الأمر Set Debug Desktop بالقائمة الفرعية للأمر View Desktops بالقائمة**

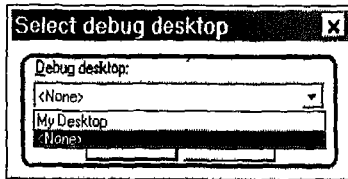
View الأمر Set Debug Desktop بالقائمة الفرعية للأمر View Desktops بالقائمة يسمح لك بأن تختار واحد من تخطيطات سطح المكتب التى قمت بحفظها على أساس أن هذا التخطيط سيتم استخدامه فى كل من مرحلة التشغيل Run-Time ومرحلة الـ debugging.

عند اختيار الأمر Set Debug Desktop من القائمة الفرعية للأمر View Desktops بالقائمة View يظهر على الشاشة صندوق الحوار Select debug desktop الموضح فى الشكل التالى :



شكل توضيحي :

صندوق الحوار Select debug desktop يعرض التخطيطات التى تستطيع أن تختار منها وذلك من خلال القائمة المنسدلة Debug desktop كما هو موضح بالشكل التالى :



شكل توضيحي :

لكى تحدد سطح المكتب الـ debug اختر سطح المكتب الذى ترغب فى استخدامه من أجل مرحلة Debugging ثم انقر بالماوس على المفتاح Ok. ومن ثم فإن سطح المكتب الـ debug يتم عرضه تلقائياً فى أثناء كافة مراحل التى تمر بها عملية الـ debugging.

عندما يخرج البرنامج من مود الـ debug ستجد أن سطح المكتب الحالى ينقلب ليصبح آخر سطح المكتب كنت تستخدمه قبل أن يدخل البرنامج فى مود الـ debug.






الأمر Toggle Form/Unit بالقائمة View

من خلال الأمر Toggle Form/Unit بالقائمة View يمكن التنقل (ذهابا وإيابا) بين الفورمة الحالية وملف الوحدة الخاص بها والذي يتم عرضه فى نافذة محرر الكود البرمجى Code Editor.

هناك طرق أخرى لأداء نفس المهمة التى يقوم بها هذا الأمر وهذه الطرق عبارة عن الآتى :

● الضغط على المفتاح F12 بلوحة المفاتيح.

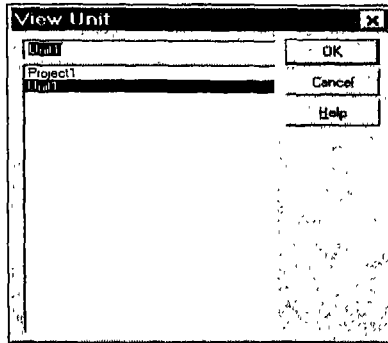
● النقر بالماوس على الأيقونة Toggle Form/Unit  بشريط الأدوات View الموضح فى الشكل التالى :




شكل توضيحى :

الأمر Units بالقائمة View

اختيار الأمر Units من القائمة View يؤدى إلى فتح صندوق الحوار View Unit الموضح فى الشكل التالى :



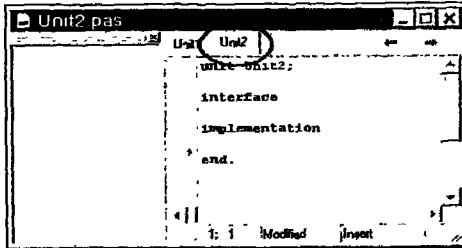
شكل توضيحى :

هناك طريقة أخرى لفتح صندوق الحوار View Unit وهى الضغط على المفاتيح Ctrl+F12 بلوحة المفاتيح. كما يمكن أيضا النقر بالماوس على الأيقونة View Unit  بشريط الأدوات View.



صندوق الحوار View Unit

يمكن استخدام صندوق الحوار View Unit لمشاهدة ملف المشروع أو ملف أى وحدة فى المشروع الحالى. هذا وعندما تفتح وحدة فإنه تصبح الصفحة النشطة فى محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

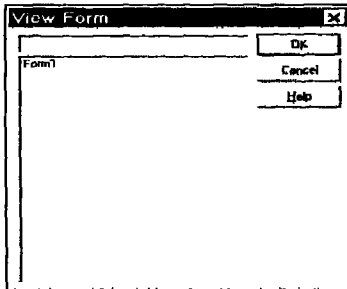


شكل توضيحي :


فى صندوق الحوار View Unit انقر بالماوس نقرا مزدوجا على اسم الوحدة التى ترغب فى عرضها على الشاشة. وإذا لم تكن الوحدة التى اخترتها هى الوحدة المفتوحة حاليا فستقوم اللغة على الفور بجعلها هى الوحدة المفتوحة حاليا.

الأمر Forms بالقائمة View

اختيار الأمر Forms من القائمة View يؤدى إلى فتح صندوق الحوار View form الموضح فى الشكل التالى :



شكل توضيحي :

هناك طريقة أخرى لفتح صندوق الحوار View form وهى الضغط على المفاتيح Shift+F12 بلوحة المفاتيح. كما يمكن أيضا النقر بالماوس على الأيقونة View Form  بشريط الأدوات View.





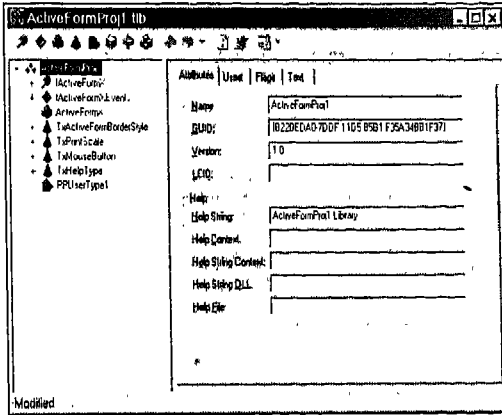
صندوق الحوار View form

يمكن استخدام صندوق الحوار View form لمشاهدة أى فورمة فى المشروع الحالى. وعندما تختار فورمة فإنها تصبح الفورمة النشطة فى المشروع كما أن ملف الوحدة الخاص بها يصبح الـ Module النشط فى نافذة محرر الكود البرمجى Code Editor.

فى صندوق الحوار View form انقر بالماوس نقرا مزدوجا على الفورمة التى ترغب فى عرضها على الشاشة وفى هذه الحالة إذا كان ملف الوحدة الخاص بهذه الفورمة غير مفتوح عندما اخترت الفورمة فستقوم اللغة على الفور بفتح هذا الملف فى محرر الكود البرمجى Code Editor.

الأمر Type Library بالقائمة View

استخدام الأمر Type Library من القائمة View يؤدي إلى فتح محرر مكتبة الأنواع Type Library الموضح فى الشكل التالى :



شكل توضيحي :

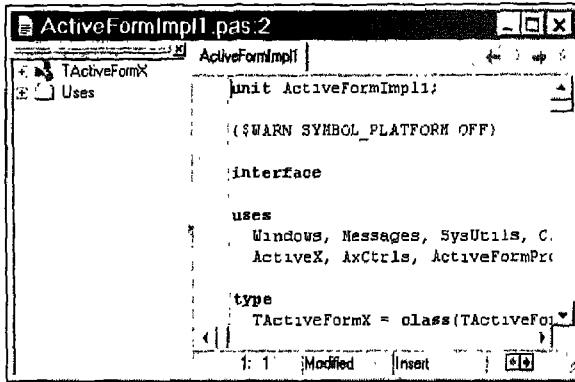
محرر مكتبة الأنواع Type Library تسمح لك بأن تختبر وتنشأ type information لكل من أدوات التحكم ActiveX والخوادم Automation والكائنات MTS والمكونات COM الأخرى. وأنت تستطيع توفير type information مع كائن إما مستقل stand-alone فى أى مكتبة من مكتبات الأنواع (المخزنة فى ملف له امتداد .TLB). كما يمكنك أيضا تكامل type information داخل ملف تنفيذى .EXE. أو مكتبة من مكتبات أدوات التحكم ActiveX على أساس أنه مصدر.

عن طريق ضم مكتبة النوع مع تطبيق أو مكتبة ActiveX فإنك تصنع معلومات مشتملة فى المكتبة مثل واجهات الاستخدام Interface للكائن الخاص بها والخصائص والأساليب والأحداث والتي تصبح متاحة من تطبيق لآخر ولكافة أدوات البرمجة.

عندما تستخدم المعالجات Wizards لإنشاء أداة تحكم ActiveX أو كائن تفعيل ذاتى Automation فإنه يتم تلقائيا إنشاء مكتبة نوع. وأنت تستطيع بعد ذلك استخدام محرر مكتبة النوع Type Library لاختبار أو تعديل type information التى تم إنشاؤها بواسطة المعالج. هذا ويمكن استخدام محرر مكتبة النوع Type Library لإضافة الصفات الوظيفية مثل خصائص جديدة أو أساليب أو أحداث لمكتبة النوع التى تعدها.

الأمر New Edit Window بالقائمة View

اختيار الأمر New Edit Window من القائمة View يؤدي إلى فتح محرر جديد للكود البرمجى كما هو موضح فى الشكل التالى :



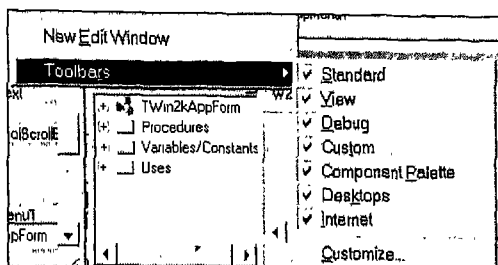
شكل توضيحي :

ونافذة محرر الكود البرمجى الجديدة تكون مشتملة على نسخة من الصفحة النشطة من محرر الكود البرمجى Code Editor الأسمى. هذا وأى تغييرات تجريها إما فى المحرر الأسمى أو فى المحرر الجديد ينعكس على محتويات كليهما. ومن ثم تستطيع التمييز بين النوافذ حيث أن يكون الاسم الظاهرى Caption للنافذة الأصلية مشتملا على الرقم (١) ويكون الاسم الظاهرى Caption لأول نسخة مشتملا على الرقم (٢) ويكون الاسم الظاهرى Caption لثانى نسخة مشتملا على الرقم (٣) وهكذا...



الأمر Toolbars بالقائمة View

يعمل الأمر Toolbars بالقائمة View على إظهار أو إخفاء شرائط الأدوات وذلك من خلال القائمة الفرعية الخاصة به والموضحة بالشكل التالى :



شكل توضيحي :

الجدول التالى يقدم لنا وصف مختصر لكل شريط من شرائط الأدوات التالية فى بيئة التطوير المتكاملة IDE :

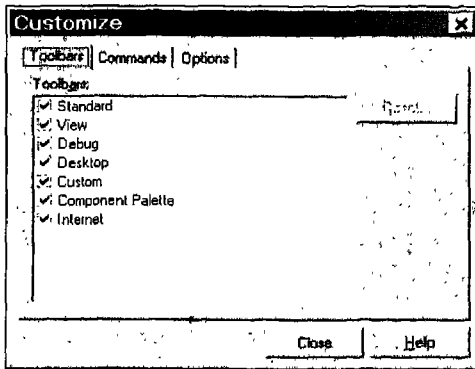
الوصف والاستخدام	شريط الأدوات
من خلال الأيقونات الموجودة فى شريط الأدوات هذا يمكن إضافة عنصر جديد للمشروع الحالة وفتح أى عنصر من العناصر التالى تم حفظها قبل ذلك وحفظ المشروع الحالى وحفظ كافة المشاريع المفتوحة حالياً وفتح مشروع من المشاريع التى سبق العمل بها وكذلك إضافة ملف إلى المشروع الحالى.	Standard
من خلال الأيقونات الموجودة فى شريط الأدوات هذا يمكن مشاهدة ملف الوحدة أو مشاهدة فورمة أو التنقل بين الفورمة وملف الوحدة أو إضافة فورمة جديدة للمشروع.	View
من خلال الأيقونات الموجودة فى شريط الأدوات هذا يمكن تشغيل المشروع أو تعطيل تشغيله بصفة مؤقتة أو تتبع الأخطاء والثغرات بالكود البرمجى.	Debug
بشريط الأدوات هذا يمكن إضافة أى أوامر ترغبها. وفى الأصل يكون شريط الأدوات هذا مشتملاً على أيقونة Help فقط.	Custom



Component Palette	شريط الأدوات هذا يتألف من عدة تبويبات تشتمل على أيقونات لكافة المكونات سواء المرئية أو الغير مرئية والتي تستخدمها لتصميم التطبيق الذى تعده.
Desktops	من خلال الأيقونات الموجودة فى شريط الأدوات هذا يمكن اختيار واحد من تخطيطات سطح المكتب المتاحة بالإضافة لحفظ التخطيط الحالى لسطح المكتب مع إمكانية جعله debug Desktop.
Internet	شريط الأدوات هذا يشتمل على أيقونات للمكونات التى تستطيع استخدامها لإنشاء تطبيقات لديها القدرة على العمل بشبكة الويب.

بالقائمة الفرعية الخاصة بالأمر Toolbars قم بالتعليم على أسماء شرائط الأدوات التى ترغب فى إظهارها على الشاشة أما شرائط الأدوات التى ترغب فى إخفاؤها من على الشاشة فقم بإلغاء التعليم من عليها بهذه القائمة الفرعية.

تستطيع أيضا تفصيل كافة شرائط الأدوات لتصبح كما ترغب أنت وهذا التفصيل يتم عن طريق إضافة أو إزالة أيقونات من شرائط الأدوات وذلك من خلال الأمر Customize بالقائمة الفرعية للأمر Toolbars والذى يعمل على فتح صندوق الحوار Customize الموضح فى الشكل التالى :



شكل توضيحي :



شريط الأدوات القياسى Standard Toolbar

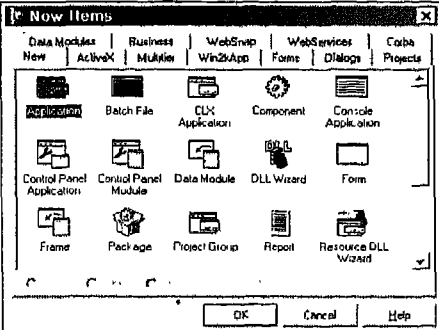

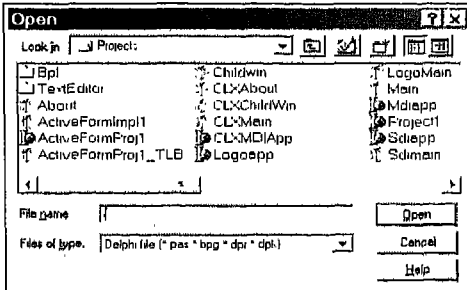

الشكل التالى يوضح لنا شريط الأدوات القياسى Standard Toolbar :



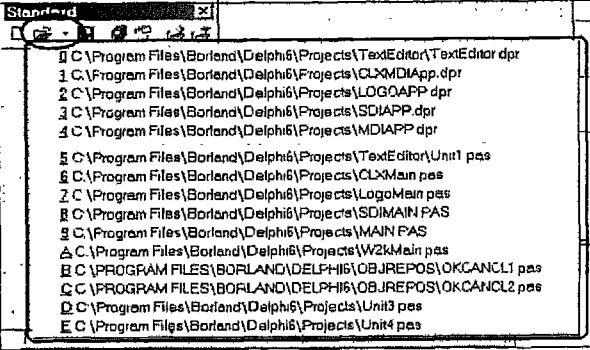



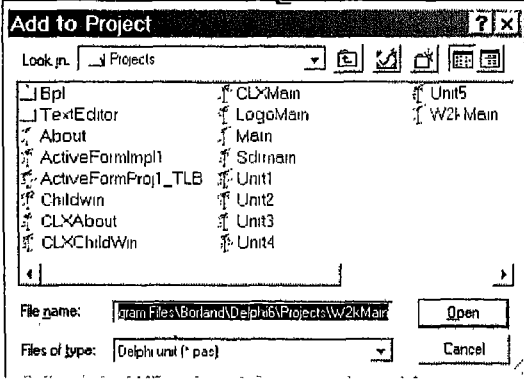


شكل توضيحي :

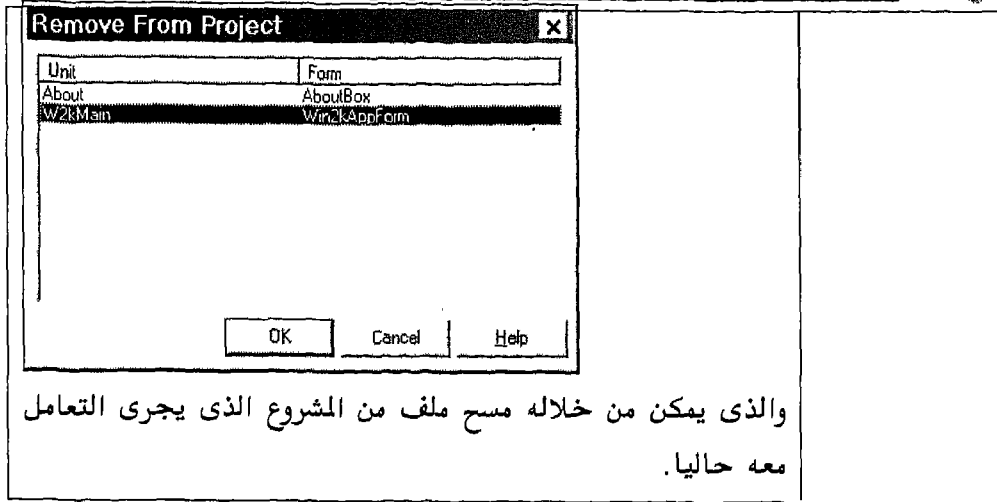
من خلال الجدول التالى نتعرف على وظيفة الأيقونات الموجودة فى شريط الأدوات

القياسى Standard Toolbar :

الوظيفة	الأيقونة
<p>فتح صندوق الحوار New Items الموضح فى الشكل التالى :</p> 	
<p>فتح صندوق الحوار File Open الموضح فى الشكل التالى :</p>  <p>وعند النقر بالماوس على السهم الموجود بهذه الأيقونة يتم فتح قائمة تضم مجموعة من المشاريع التى سبق التعامل معه كما هو موضح بالشكل التالى :</p>	



	
<p>يسمح لك بأن تخزن التغييرات التي أجريتها على كافة الملفات الموجودة بالمشروع الذي يتم التعامل معه حاليا باستخدام الاسم الحالي لكل ملف (كما يمكن أيضا الضغط على المفاتيح Ctrl+S بلوحة المفاتيح).</p>	
<p>يسمح لك بحفظ كافة الملفات المفتوحة حاليا بما فيها ملفات المشروع الحالي والـ modules (كما يمكن أيضا الضغط على المفاتيح Shift+Ctrl+S بلوحة المفاتيح).</p>	
<p>يسمح لك بأن تفتح مشروع من المشروعات التي سبق التعامل معها (كما يمكن أيضا الضغط على المفاتيح Ctrl+F11 بلوحة المفاتيح).</p>	
<p>يفتح صندوق الحوار Add to Project الموضح في الشكل التالي :</p>  <p>والذي من خلاله يمكن إضافة ملف إلى المشروع الذي يجري التعامل معه حاليا (كما يمكن أيضا الضغط على المفاتيح Shift+F11 بلوحة المفاتيح).</p>	
<p>يفتح صندوق الحوار Remove from Project الموضح في الشكل التالي :</p>	



شريط الأدوات View Toolbar

الشكل التالى يوضح لنا شريط الأدوات View Toolbar :

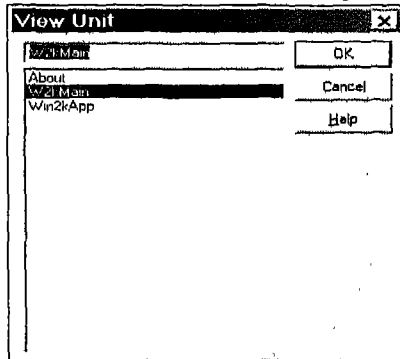


شكل توضيحي :

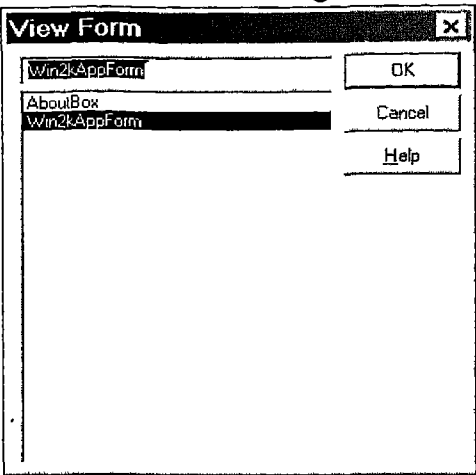



من خلال الجدول التالى نتعرف على وظيفة الأيقونات الموجود فى شريط الأدوات

: View Toolbar

الأيقونة	الوظيفة
	يفتح صندوق الحوار View Unit الموضح فى الشكل التالى :





والذى يكن من خلاله مشاهد محتويات وحدة من الوحدات الموجودة بالمشروع (كما يمكن أيضا الضغط على المفاتيح Ctrl+F12 بلوحة المفاتيح).	
<p>يفتح صندوق الحوار View Form الموضح فى الشكل التالى :</p>  <p>والذى يمكن من خلاله عرض فورمة من الفورم الموجودة بالمشروع (كما يمكن أيضا الضغط على المفاتيح Shift+F12 بلوحة المفاتيح).</p>	
التنقل بين الفورمة وملف الوحدة الخاصة بها فى محرر الكود البرمجى Code Editor (كما يمكن أيضا الضغط على المفتاح F12 بلوحة المفاتيح).	
إضافة فورمة جديدة إلى المشروع.	

شريط الأدوات Debug Toolbar

الشكل التالى يوضح لنا شريط الأدوات Debug Toolbar :



شكل توضيحي :



من خلال الجدول التالى نتعرف على وظيفة الأيقونات الموجود فى شريط الأدوات

: Debug Toolbar

الأيقونة	الوظيفة
	تشغيل وترجمة المشروع الذى يتم العمل به حاليا وعند النقر بالماوس على السهم الموجود بهذه الأيقونة يتم فتح قائمة تضم مجموعة من المشاريع المفتوحة حاليا لكى تختار منها كما هو موضح بالشكل التالى :
	وهذا الأمر يناظر الأمر Run الموجود بالقائمة Run (يمكن أيضا الضغط على المفتاح F9 بلوحة المفاتيح).
	تعطيل تشغيل المشروع بصفة مؤقتة وهذا الأمر يناظر الأمر Program Pause الموجود بالقائمة Run.
	تتبع الأخطاء والثغرات فى الكود البرمجى سطر بسطر كما هو موضح بالشكل التالى :



(يمكن أيضا الضغط على المفتاح F7 بلوحة المفاتيح) وهذا الأمر ينظر الأمر Trace Into الموجود بالقائمة Run	
تخطى خطوة في أثناء تتبع الأخطاء والثغرات في الكود البرمجي (يمكن أيضا الضغط على المفتاح F8 بلوحة المفاتيح).	

شريط الأدوات Custom Toolbar

الشكل التالي يوضح لنا شريط الأدوات Custom Toolbar :



شكل توضيحي :

من خلال الجدول التالي نتعرف على وظيفة الأيقونات الموجود في شريط الأدوات

: Custom Toolbar

الأيقونة	الوظيفة
	فتح صندوق الحوار Help الموضوع في الشكل التالي :

بالرغم من أن شريط الأدوات Custom Toolbar يشتمل على الأيقونة Help فقط إلا إنه من الممكن إضافة أى أيقونات لشريط الأدوات هذا.





شريط الأدوات Desktops Toolbar

الشكل التالى يوضح لنا شريط الأدوات Desktops Toolbar:



شكل توضيحى :

من خلال الجدول التالى نتعرف على وظيفة الأيقونات الموجود فى شريط الأدوات

:Desktops Toolbar

الأيقونة	الوظيفة
	تسمح لك بأن تتحول إلى أى من التخطيطات المتعددة لسطح المكتب والتي تم إنشاؤها وحفظها من قبل.
	عرض صندوق الحوار Save Desktop الموضح فى الشكل التالى :
	والذى يمكن من خلاله تحديد الاسم الذى ترغب فى استخدامه لحفظ تخطيط سطح المكتب الذى أنشأته.
	جعل التخطيط الحالى لسطح المكتب هو نفس تخطيط سطح المكتب عندما يكون البرنامج فى المود Debug أو عندما يكون فى مرحلة التشغيل Run-Time.

شريط الأدوات Internet Toolbar

الشكل التالى يوضح لنا شريط الأدوات Internet Toolbar:

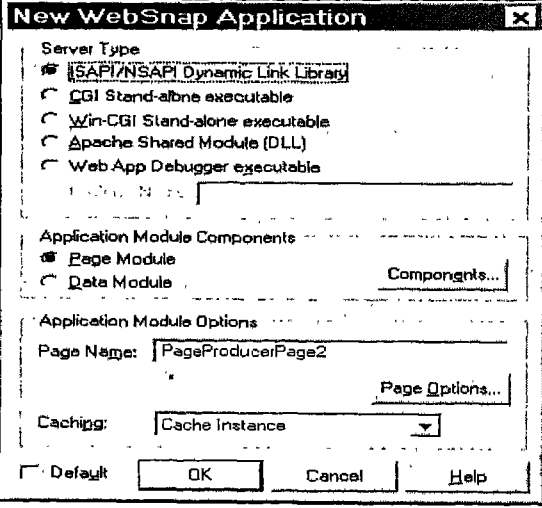




شكل توضيحى :



من خلال الجدول التالى نتعرف على وظيفة الأيقونات الموجودة فى شريط الأدوات

: Internet Toolbar

الوظيفة	الأيقونة
<p>يعرض صندوق الحوار New WebSnap Application الموضح فى الشكل التالى :</p>  <p>وصندوق الحوار هذا يعتبر أول خطوة فى عملية إنشاء تطبيق يعمل فى خوادم شبك الويب Web Server مع Web Snap كما يمكن استخدام نفس صندوق الحوار هذا فى عملية تحويل أى تطبيق سبق إنشاؤه ليصبح من النوع الذى يمكن أن يعمل بخوادم شبكة الويب.</p>	 <p>New WebSnap Application</p>
<p>يعرض صندوق الحوار WebSnap Page Module الموضح فى الشكل التالى :</p>	 <p>New WebSnap Page Module</p>



يمكن استخدام صندوق الحوار هذا لإنشاء صفحة محتوى. ونود هنا القول بأن Module الصفحة يكون مشتملا على الأداة page producer التي تكون مسئولة عن تكوين محتويات أى صفحة.

عرض صندوق الحوار New WebSnap Data Module الموضح في الشكل التالي :

من خلال صندوق الحوار هذا يتم إنشاء Module بيانات فارغ بحيث يتم فيه إضافة المكونات الخاصة بـ WebSnap Internet. وفي هذا الصدد نقول إن اختيار هذا النوع من Module لا يؤدي إلى إنشاء صفحة محتوى. وهذا Module يتم استخدامه كما لو كان حاوية للمكونات المتاحة للاستخدام المشترك بواسطة Modules الأخرى- كمكونات قواعد البيانات المستخدمه من خلال الإثنين.

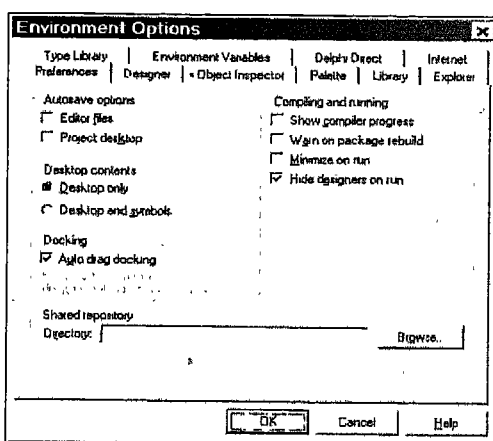

New
WebSnap
Data
Module

**استخدام المحرر الخارجي External Editor**

لكي تتمكن من التوجه إلى المحرر الخارجي External Editor ينبغي عليك أولاً تهيئة وإعداده.

خطوات إعداد وتهيئة المحرر الخارجي External Editor

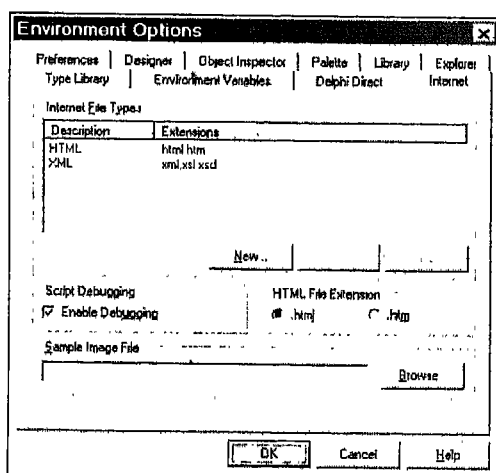
(١) افتح القائمة Tools واختر منها الأمر Environment Options ليظهر على الشاشة صندوق الحوار



شكل توضيحي :

(٢) انقر بالماوس على التبويب Internet ليظهر على السطح كما هو موضح بالشكل

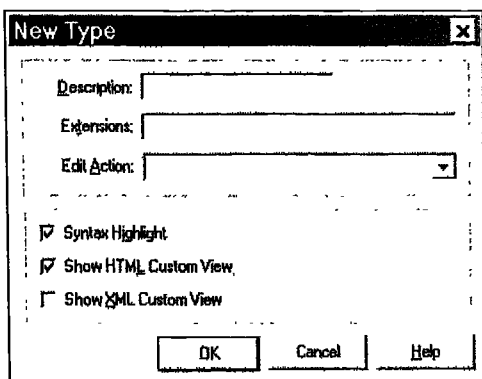
التالي :



شكل توضيحي :



(٣) انقر بالماوس على المفتاح New ليظهر على الشاشة صندوق الحوار New Type الموضح فى الشكل التالى :




شكل توضيحى :

(٤) بالحقل Description اكتب وصف مختصر للمحرر الخارجى ثم فى الحقل Extensions اكتب امتدادات الملفات التى ترغب فى فتحها من خلال المحرر الخارجى

(٥) فى القائمة المنسدلة Edit Action التى تعرض أفعال التحرير المتاحة والمخزنة فى الـ Registry الخاص بجهازك أو امتدادات الملفات التى أدخلتها فى الحقل Extensions.

الأفعال التقليدية عبارة عن الفعل Edit والفعل Edit with Notepad والفعل Open ولكن هناك العديد من الأفعال الأخرى والتى يمكن عرضها فى القائمة المنسدلة Edit Action وذلك بناء على تطبيقات المركبة فى جهاز الكمبيوتر الخاص بك.




(٦) اختر الفعل Edit الذى ترغب فى استخدامه مع الملفات عند التعامل مع الأمر External Editor بالقائمة Tools أو عند النقر بالماوس على الأيقونة  بشريط الأدوات Internet External Editor.

(٧) انقر بالماوس على المفتاح Ok لغلق صندوق الحوار New Type.

(٨) انقر بالماوس على المفتاح Ok لغلق صندوق الحوار Environment Options.

**خطوات فتح المحرر الخارجى External Editor**

بمجرد أن تفتح ملف ينتمى لأنواع الملفات التى حددتها من خلال صندوق الحوار New Type تستطيع إذا أن تفتح المحرر الخارجى External Editor من خلال الخطوات التالية :

- (١) اختر الأمر External Editor من القائمة Tools أو انقر بالماوس على الأيقونة  بشريط الأدوات Internet.
- (٢) ستجد أن تطبيق الويب الذى تتعامل معه قد قام بفتح المحرر الذى قمت بتركيبه قبل ذلك.

القائمة Project Menu

يمكن استخدام القائمة Project لترجمة أو بناء التطبيق الذى تعده. هذا وينبغى أن يكون لديك مشروع واحد على الأقل مفتوح.
من خلال الجدول التالى سنتعرف سويا باختصار على وظيفة كل أمر من الأوامر الموجودة فى القائمة Project :

الأمر	الوصف والاستخدام
Add to Project	إضافة ملف إلى المشروع الحالى.
Remove from project	إزالة ملف من المشروع الحالى.
Import Type Library	استيراد واحدة من مكتبات الأنواع للمشروع الذى يتم التعامل معه حالياً.
Add to Repository	إضافة المشروع الذى ترغبه إلى مخزن الكائنات Object Repository.
View Source	عرض ملف المشروع فى نافذة محرر الكود البرمجى Code Editor.
Languages	يسمح لك بأن تضيف أو تلغى أو تقوم بتحديث الملفات DLL المصدرية أو تختار لغة من أجل الاختبار.



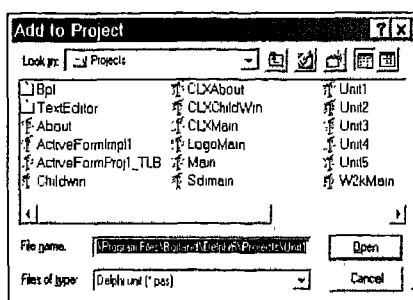
فتح صندوق الحوار New Items الذى يحتوى على عدد من المعالجات والكائنات المخزنة فى مخزن الكائنات. وانت تستطيع إما أن تقوم بتكوين كائن جديد أو البدء مع أى كائن من الكائنات السابقة الإعداد والمخزنة فى مخزن الكائنات.	Add New Project
من خلال هذا الأمر يمكن استخدام صندوق الحوار Open Project لإضافة مشروع موجود بالفعل إلى مدير المشروع.	Add Existing Project
يقوم هذا الأمر بترجمة الملفات الموجودة فى المشروع الحالى فقط والتي حدث لها تغييرات منذ أن تم بناؤها آخر مرة.	Compile project
ترجمة أى عنصر فى المشروع بغض النظر عما إذا كان حدث تغيير فى المصدر أم لا.	Build project
ترجمة المشروع الذى تعده ولكنه لا يقوم بربطه مع مكتبات اللغة.	Syntax Check project
عرض كافة معلومات البناء و حالته بالنسبة للمشروع الذى تتعامل معه.	Information for project
ترجمة أى كود برمجى مصدرى تم تغييره منذ آخر ترجمة فى كافة المشروعات الموجودة فى مجموعة المشروع.	Compile All Projects
ترجمة أى شئ فى مجموعة المشروع بغض النظر عن عما إذا كان حدث تغيير فى الأصول أم لا.	Build All Projects
إعداد القيم التحديدية Settings الضرورية لنشر وتوزيع أدوات التحكم ActiveX أو الفورم Active Form التى تم الإنتهاء من إعدادها.	Web Deployment Options
بعد تحديد الخيارات الخاصة بالنشر عبر شبكة الويب وترجمة المشروع يقوم هذا الأمر بنشر أدوات التحكم ActiveX أو الفورم Active Form التى إنتهيت من إعدادها	Web Deploy



Options	عرض صندوق الحوار Project Options والذي يمكن من خلاله التعامل مع الاختيارات الخاصة بك من عملية الترجمة وعملية الربط والفورم الافتراضية ومعلومات الإصدار...بالإضافة إلى العديد من العناصر الأخرى.
---------	---

الأمر Add to Project بالقائمة Project

اختيار الأمر Add to Project من القائمة Project يؤدي إلى فتح صندوق الحوار Add To Project الموضح في الشكل التالى :



شكل توضيحي :

يمكن أيضا فتح صندوق الحوار Add To Project عن طريق الضغط على المفاتيح Shift+F11 بلوحة المفاتيح.



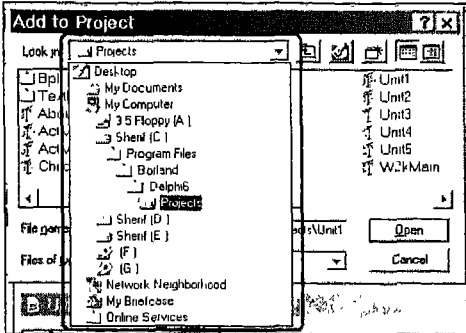
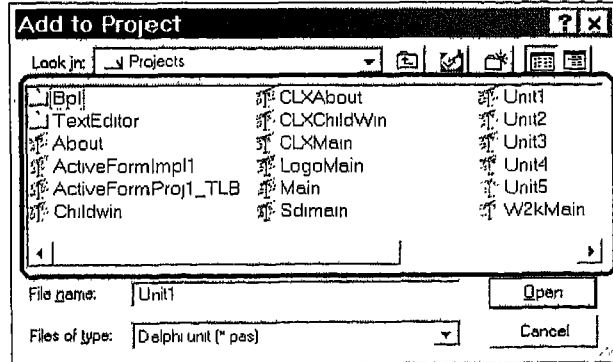
صندوق الحوار Add To Project

يمكن استخدام صندوق الحوار Add To Project لإضافة وحدة من الوحدات الموجودة بالفعل كما يتم أيضا إضافة الفورمة الملحقة بالوحدة المراد إضافتها إلى المشروع الذى تعمل به حاليا. هذا وعندما تضيف وحدة إلى مشروع تجد أن اللغة تقوم تلقائيا بإضافة هذه الوحدة إلى الجملة uses بملف المشروع.

الجدول التالى يقدم لنا وصف مختصر للأجزاء التى يتألف منها صندوق الحوار

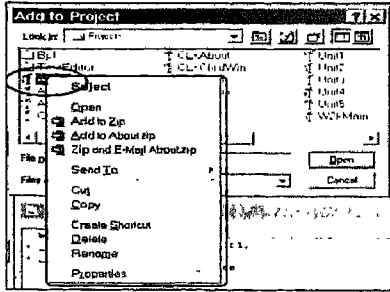
: Add To Project



الوصف والاستخدام	المعامل
<p>من خلال هذه القائمة المنسدلة نستطيع اختيار مشغل الأقراص أو الفهرس الذى سيتم البحث فيه عن الملفات المطلوب إضافتها للمشروع كما هو موضح بالشكل التالى :</p>  <p>بعد ذلك نعرض نافذة الملف الفهارس الفرعية الموجودة داخل مشغل الأقراص المختار أو داخل الفهرس المختار كما إنها تعرض أيضا كافة الملفات التى تتطابق أنواعها مع الأنواع المحددة بالفلتر الموجود بالقائمة المنسدلة Files of Type.</p>	<p>Look In</p>
<p>نافذة الملف تعرض الفهارس الفرعية الموجودة بالفهرس المفتوح حاليا بالإضافة إلى الملفات الموجودة فى الفهرس المفتوح حاليا والتى تتطابق مع الفلتر المطبق حاليا بالقائمة المنسدلة Files of Type كما هو موضح بالشكل التالى :</p> 	<p>File Window</p>

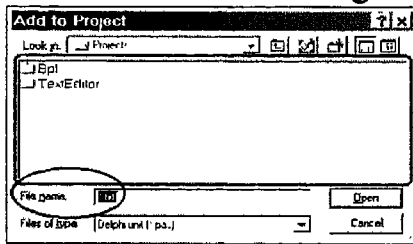


اختر الملف الذي ترغب فيه، إضافته إلى المشروع وذلك عن طريق النقر عليه بالماوس أو باستخدام مفاتيح الأسهم الموجودة فيه، لوحة المفاتيح أو بالنقر بالزر الأيمن للماوس على الملفات التي ترغبها وذلك لعرض القائمة المختصرة الموضحة في الشكل التالي :



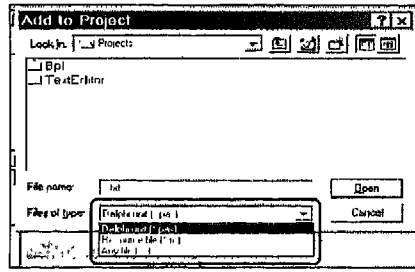
ومنها اختر الأمر Select.

بهذا الحقل يمكن أن تكتب اسم الملف الذي ترغب في إضافته للمشروع كما يمكن أيضا أن تكتب فلتر *.ext للحد من الملفات التي تظهر في نافذة الملف كما هو موضح بالشكل التالي :





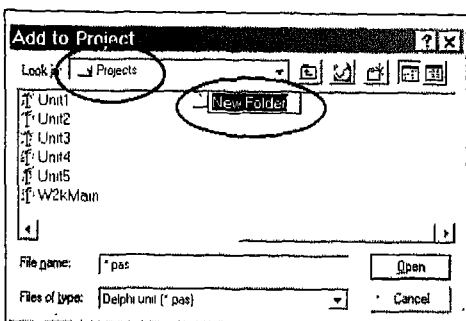


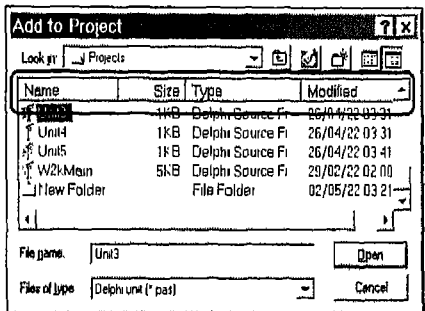
File Name

من خلال هذه القائمة المنسدلة تستطيع أن تختار الفلتر الذي يعمل على تحديد نوعية الملفات التي تظهر في المنطقة File Window. هذا والشكل التالي يوضح لنا الفلاتر الموجودة بهذه القائمة المنسدلة :



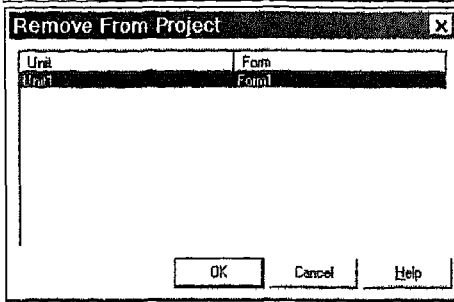
Files of type



<p>النقر على هذه الأيقونة  يؤدي إلى الانتقال للمستوى الأعلى فى الهيكل الشجرى الموجود به الفهرس المفتوح حاليا.</p>	<p>Up on Level</p>
<p>النقر على هذه الأيقون  يعمل على إنشاء فهرس جديد ليكون فهرس فرعى بالفهرس أو مشغل الأقراص المفتوح حاليا كما هو موضح بالشكل التالى :</p> 	<p>Create New Folder</p>
<p>هذه الأيقونة  تعمل على عرض أسماء الملفات فقط فى المنطقة File Window وهذا هو الوضع الطبيعى لطريقة العرض.</p>	<p>List</p>
<p>هذه الأيقونة  تعمل على عرض معلومات تفصيلية عن كل عنصر موجود فى المنطقة File Window كالأسم والحجم كما هو موضح بالشكل التالى :</p> 	<p>Details</p>

الأمر Remove from Project بالقائمة Project

اختيار الأمر Remove from Project من القائمة Project يؤدي إلى فتح صندوق الحوار Remove from project الموضح فى الشكل التالى :

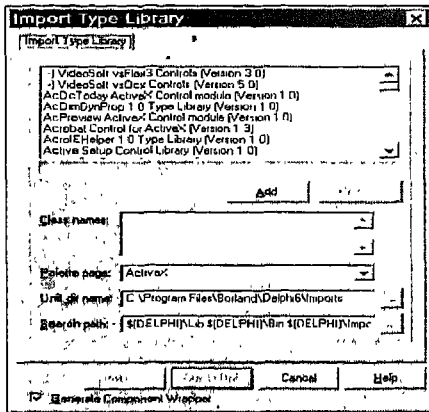


شكل توضيحي :

من خلال صندوق الحوار Remove from project تستطيع اختيار العنصر الذى ترغب فى إزالته من المشروع ثم النقر بالماوس على المفتاح Ok.

الأمر Import Type Library بالقائمة Project

اختيار الأمر Import Type Library من القائمة Project يؤدي إلى فتح صندوق الحوار Import Type Library الموضح فى الشكل التالى :

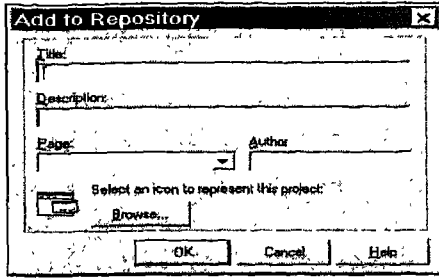


شكل توضيحي :

من خلال صندوق الحوار Import Type Library تستطيع اختيار مكتبة النوع التى ترغب فى استيرادها ثم تنقر بالماوس على المفتاح Create Unit.

الأمر Add To Repository بالقائمة Project

اختيار الأمر Add To Repository من القائمة Project يؤدي إلى فتح صندوق الحوار Add To Repository الموضح فى الشكل التالى :

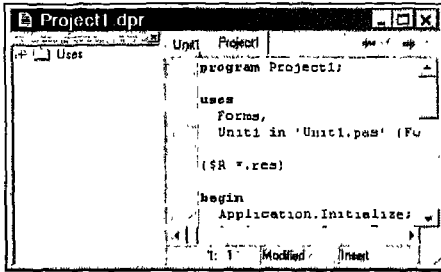


شكل توضيحي :

عن طريق صندوق الحوار Add To Repository يمكن أن تضيف المشاريع والفورم التي ترغبها إلى مخزن الكائنات.

الأمر View Source بالقائمة Project

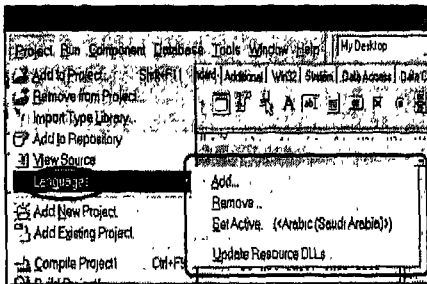
اختيار الأمر View Source من القائمة Project يؤدي إلى عرض ملف المشروع للمشروع الحالي وذلك في نافذة محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالي :



شكل توضيحي :

الأمر Languages بالقائمة Project

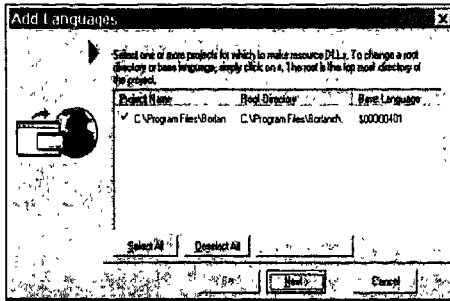
القائمة الفرعية الخاصة بالأمر Languages بالقائمة Project والموضحة في الشكل التالي تشتمل على عدد من الاختيارات التي تعمل على تغيير القيم التحديدية الخاصة بترجمة المشروع الذي تتعامل معه :



شكل توضيحي :

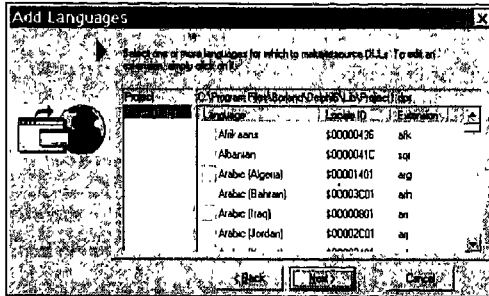
**الأمر Add بالقائمة الفرعية للأمر Languages بالقائمة Project**

يعمل الأمر Add بالقائمة الفرعية للأمر Languages بالقائمة Project على فتح المعالج Add Languages وفيه يظهر أول صندوق حوار كما هو موضح بالشكل التالي :



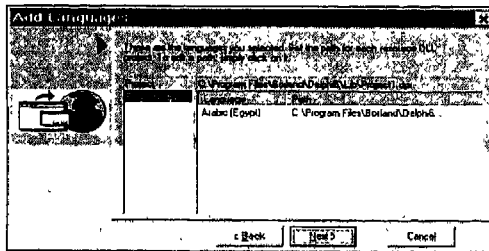
شكل توضيحي :

في صندوق الحوار هذا تختار واحد أو أكثر من المشروعات التي تصنع مكتبات الإتصال الديناميكي DLLs وفيه انقر بالماوس على المفتاح Select All ثم على المفتاح Next للانتقال لصندوق الحوار التالي والموضح في الشكل التالي :



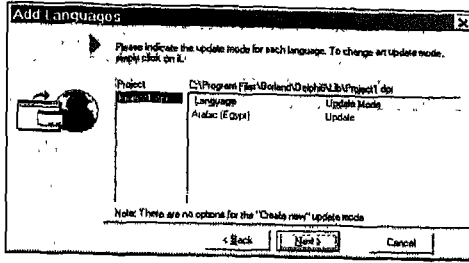
شكل توضيحي :

في صندوق الحوار هذا تستطيع أن تختار لغة واحد أو أكثر لاستخدامها في إعداد المكتبات DLL وفيه علم على المربع المجاور للغة Arabic (Egypt) ثم انقر بالماوس على المفتاح Next للانتقال لصندوق الحوار التالي والموضح في الشكل التالي :



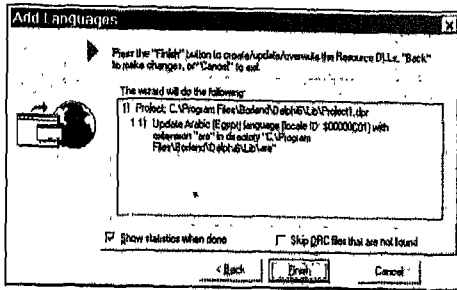
شكل توضيحي :

بصندوق الحوار هذا تشاهد اللغات التى اخترتها والمطلوب منك الآن تحديد مسار لكل ملف من الملفات DLLs ثم انقر بالماوس على المفتاح Next للانتقال لصندوق الحوار التالى والموضح فى الشكل التالى :



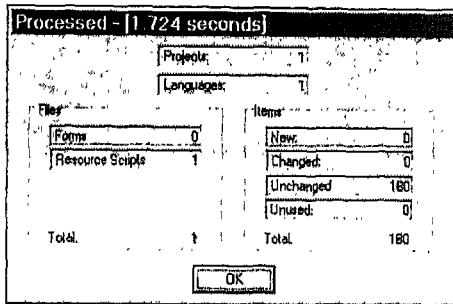
شكل توضيحي :

صندوق الحوار هذا يطلب منك تحديد مود التحديث لكل لغة وبعد ذلك انقر بالماوس على المفتاح Next للانتقال لصندوق الحوار التالى والموضح فى الشكل التالى :



شكل توضيحي :

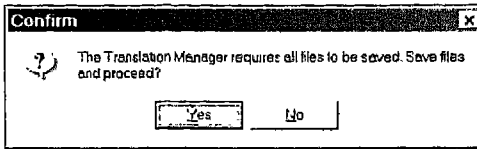
يطلب منك صندوق الحوار هذا أن تنقر بالماوس على المفتاح Finish للبدء فى إنشاء الملفات DLLs للمشروع الذى تعمل به. بعد ذلك يظهر على الشاشة صندوق الحوار Processed الذى يقدم لك تقرير مختصر عن عدد الملفات والعناصر التى تم التعامل معها فى أثناء الإنشاء كما هو موضح بالشكل التالى :



شكل توضيحي :

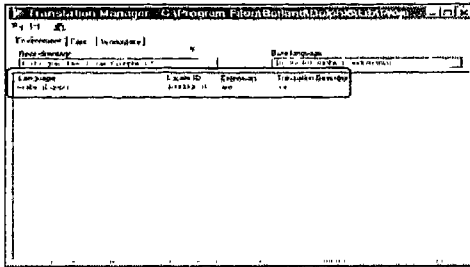


بعد أن تنتهي من قراءة التقرير انقر بالماوس على المفتاح Ok لتظهر الرسالة الموضحة في الشكل التالي :



شكل توضيحي :

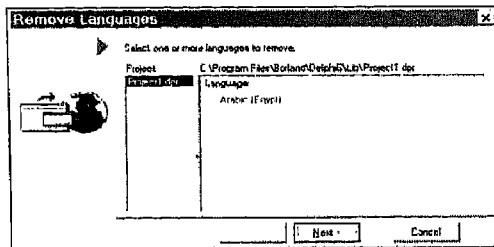
هذه الرسالة تقول لك أن مدير الترجمة يطلب منك ضرورة أن تقوم بحفظ كافة الملفات فهل أنت موافق على القيام بذلك. انقر بالماوس على المفتاح Yes. والآن تظهر على الشاشة نافذة مدير الترجمة وبها اللغة التي اخترتها كما هو موضح بالشكل التالي :



شكل توضيحي :

الأمر Remove بالقائمة الفرعية للأمر Languages بالقائمة Project

يعمل الأمر Remove بالقائمة الفرعية للأمر Languages بالقائمة Project على فتح المعالج Remove Languages وفيه يظهر أول صندوق حوار كما هو موضح بالشكل التالي :

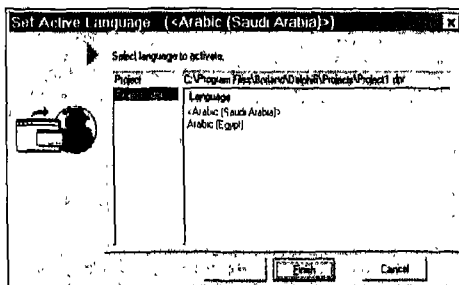


شكل توضيحي :

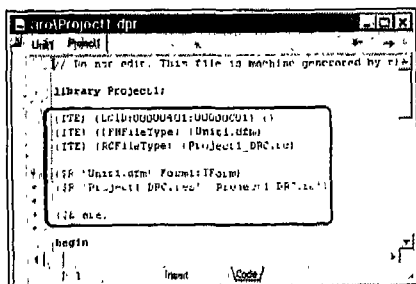
صندوق الحوار هذا يطلب منك أن تختار واحدة أو أكثر من اللغات المطلوب إزالتها وفيه علم بالماوس على اللغة Arabic (Egypt) ثم انقر بالماوس على المفتاح Next للانتقال لصندوق الحوار التالي والموضح في الشكل التالي :

بصندوق الحوار هذا انقر بالماوس على المفتاح Finish ليتم مسح الملفات DLLs الخاصة باللغة المطلوب إزالتها وللقيام أيضا بتحديث المشروع.

من خلال الأمر Set Active القائمة الفرعية للأمر Languages بالقائمة Project يعمل على يمكن اختيار DLL مصدرى من أجل الاختبار. هذا وعند استخدام هذا الأمر يظهر على الشاشة صندوق الحوار Set Active Language الموضح فى الشكل التالى :



صندوق الحوار هذا يطلب منك اختيار اللغة التي ترغب في تفعيلها وفيه اختر Arabic (Egypt) ثم انقر بالماوس على المفتاح Finish. بعد ذلك ستلاحظ حدوث بعض التغييرات في الكود البرمجي الموجود في محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالي :

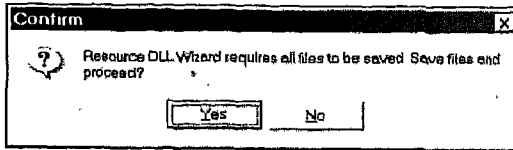


۲۲۳



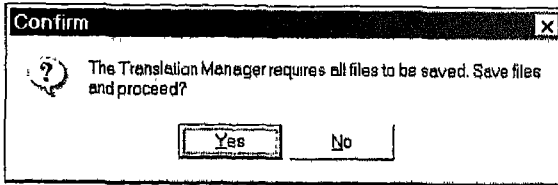
الأمر Update Resource DLLs بالقائمة الفرعية للأمر Languages بالقائمة Project

من خلال الأمر Update Resource DLLs بالقائمة الفرعية للأمر Languages بالقائمة Project يمكن تحديث الملفات DLLs التي تعتبر أحد مصادر المشروع. هذا وعند التعامل مع هذا الأمر تظهر الرسالة الموضحة في الشكل التالي :



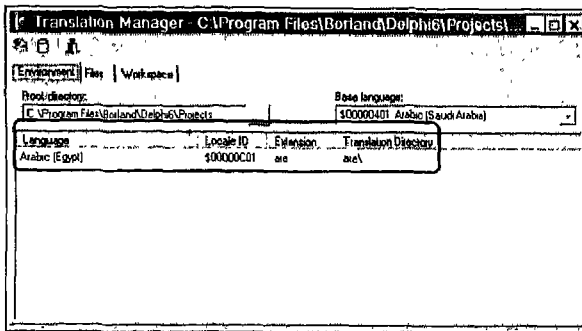
شكل توضيحي :

هذه الرسالة تقول لك أن المعالج Resource DLL Wizard يحتاج لأن يتم حفظ كافة الملفات المفتوحة الآن ومن ثم انقر بالماوس على المفتاح Yes. بعد ذلك تظهر الرسالة الموضحة في الشكل التالي :



شكل توضيحي :

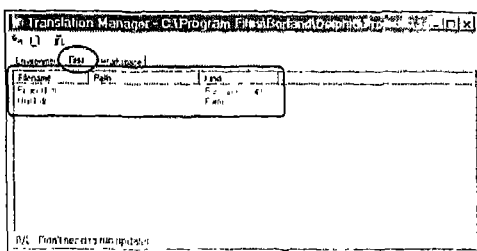
هذه الرسالة تقول لك أن مدير الترجمة Translation Manager يطلب أن يتم حفظ كافة الملفات المفتوحة كلها ومن ثم انقر بالماوس على المفتاح Yes. الآن تظهر على الشاشة نافذة مدير الترجمة Translation Manager وبها شاهد ثلاثة تبويبات هي : تبويب بيئة العمل Environment وفيه شاهد اللغة التي اخترناها وبعض المعلومات عنها كما هو موضح بالشكل التالي :



شكل توضيحي :

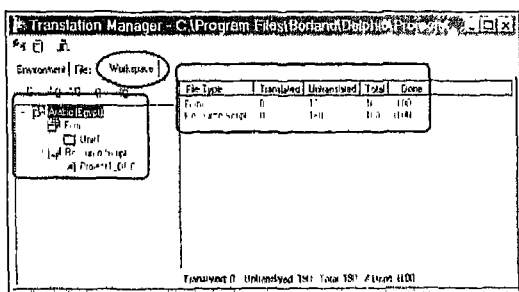


أما التبويب Files فيعرض الملفات الخاصة بالمشروع ونوعية كل منها كما هو موضح بالشكل التالي :



شكل توضيحي :

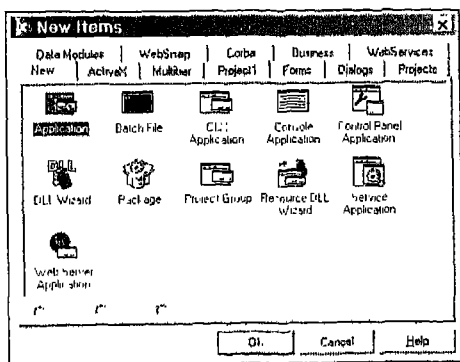
وأخيرا فتبويب منطقة العمل Workspace فيعرض كيفية ارتباط اللغة المختارة مع ملفات المشروع كما هو موضح بالشكل التالي :



شكل توضيحي :

الأمر Add New Project بالقائمة Project

يتم استخدام الأمر Add New Project من القائمة Project لإضافة عنصر جديد (مثل تطبيق جديد أو حزمة برمجية أو ملف DLL...) إلى مجموعة المشروع. ويعمل هذا الأمر على فتح صندوق الحوار New Items الموضح في الشكل التالي :



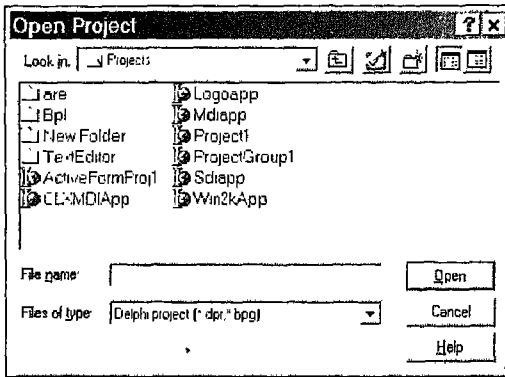
شكل توضيحي :



من خلال صندوق الحوار هذا يمكن أن تختار نوعية المشروع الذى ترغب فى إضافته إلى مجموعة المشروع.

الأمر Add Existing Project بالقائمة Project

يعمل الأمر Add Existing Project بالقائمة Project على إضافة مشروع من المشروعات التى سبق إعدادها إلى مجموعة المشروع الحالى وذلك من خلال صندوق الحوار Open Project الموضح فى الشكل التالى :

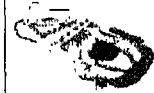


شكل توضيحي :

الأمر Compile Project بالقائمة Project

يعمل الأمر Compile Project بالقائمة Project على ترجمة كافة الملفات الموجودة فى المشروع الحالى والتى حدث بها تغيير منذ آخر بناء لها وهذه الترجمة ينتج عنها عدة أنواع من الملفات مثل الملفات التنفيذية EXE وملفات الربط الديناميكي DLLs وملفات المصادر RES وغيرها.

هناك طريقة أخرى لاستدعاء الأمر Compile Project بالقائمة Project وهى الضغط على المفاتيح Ctrl+F9 بلوحة المفاتيح.



الأمر Build Project بالقائمة Project

اختيار الأمر Build Project من القائمة Project يؤدي إلى بناء (أو إعادة بناء) كافة ملفات المشروع الحالى بغض النظر عما إذا كان حدث لها تغيير أم لا منذ آخر ترجمة



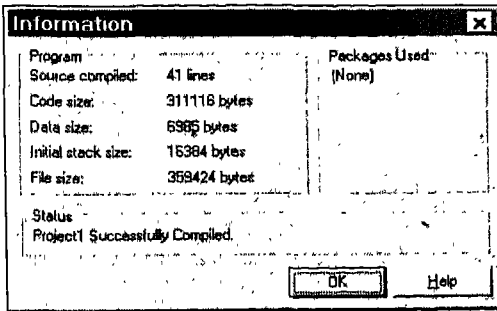
وبناء لها. وهذا الأمر يكون مفيد لك خاصة عندما تقوم بتغيير الموجهات العامة الخاصة بمترجم اللغة أو إذا كنت أجريت تغييرات على خيارات المترجم وفى هذه الحالة يجعلك هذا الأمر تتأكد من إن كافة الأكواد البرمجية قد تأثرت فعلا بهذه التغييرات.

الأمر Syntax check Project بالقائمة Project

اختيار الأمر Syntax check Project من القائمة Project يؤدي إلى ترجمة ال Modules الخاصة بالمشروع الذى تتولى إعداده ولكنه لا يقوم بربطها. ومن خلال هذا الأمر تتوفر لك الوسائل التى تمكنك من مراجعة الكود البرمجى الذى تكتبه.

الأمر Information for Project بالقائمة Project

اختيار الأمر Information for Project من القائمة Project يؤدي إلى فتح صندوق الحوار Information الموضح فى الشكل التالى :



شكل توضيحي :

من خلال صندوق الحوار هذا نحصل على معلومات حول المشروع الذى يتم العمل به وهذه المعلومات تتمثل فى عدد سطور الكود البرمجى وحجمه بالبايت وحجم البيانات بالبايت وحجم الملف وحالة ترجمة المشروع.

الأمر Information for Project لا يكون متاح للاستخدام إلا بعد أن يتم بناء أو ترجمة المشروع من خلال الأمر Build Project أو الأمر Compile Project.



**الأمر Compile All Projects بالقائمة Project**

يعمل الأمر Compile All Projects بالقائمة Project على ترجمة الملفات الموجودة بكافة المشاريع الموجودة بمجموعة المشروع علما بأنه يتم ترجمة الملفات التى حدث لها تغيير منذ آخر عملية بناء. ونود هنا القول بأن الأمر Compile All Projects يشبه إلى حد كبير الأمر Build All Projects والفرق الوحيد بينهما يتمثل فى أن الأمر Build All Projects يقوم بترجمة كافة الملفات بغض النظر عما إذا كان حدث لها تغيير أم لا.

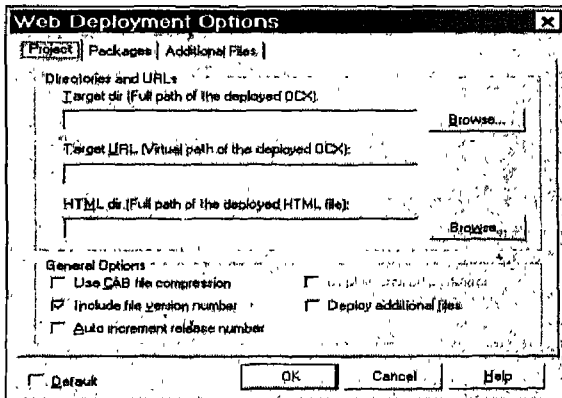
الأمر Build All Projects بالقائمة Project

يعمل الأمر Build All Projects بالقائمة Project على ترجمة الملفات الموجودة بكافة المشاريع الموجودة بمجموعة المشروع بغض النظر عما إذا كان حدث لها تغيير أم لا.

الأمر Web deployment Options بالقائمة Project

من خلال الأمر Web deployment Options يمكن تهيئة أدوات التحكم ActiveX أو الفورم التى من النوع ActiveForm التى تم الإنتهاء من إعدادها وهذه التهيئة بهدف جعل هذه العناصر متاحة للاستخدام فى خوادم شبكة الويب.

هذه التهيئة تنقسم إلى ثلاثة أقسام : قسم المشروع Project والذى يتم تهيئته من خلال التبويب Project بصندوق الحوار Web Deployment Options والموضح فى الشكل التالى :

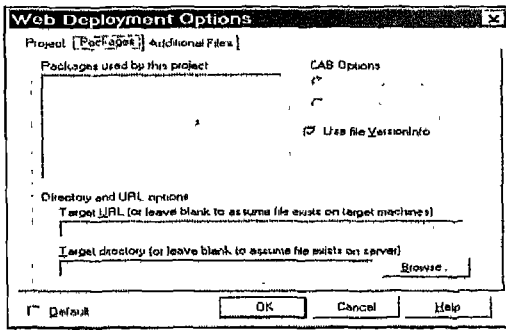


شكل توضيحي :



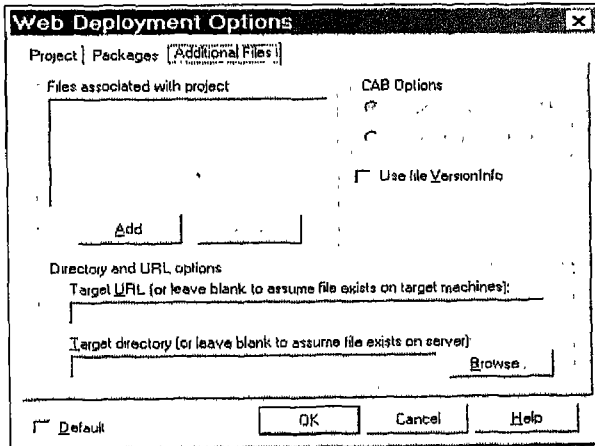
بهذا القسم يتحدد مواضع الملفات والعنوان URL لخادم الويب. كما إنه يسمح لك بأن تتعامل مع القيم التحديدية الخاصة بعملية التهيئة للملفات المضغوطة CAB والمعلومات الخاصة بالإصدار Version Information.

أما القسم الثانى فقسم الحزم البرمجية Packages التى يستخدمها المشروع والذى يتم تهيئته من خلال التبويب Packages بصندوق الحوار Web Deployment Options كما هو موضح بالشكل التالى :



شكل توضيحي :

وأخيرا قسم الملفات الإضافية المرتبطة بهذا المشروع والذى يتم تهيئته من خلال التبويب Additional Files بصندوق الحوار Web Deployment Options كما هو موضح بالشكل التالى :



شكل توضيحي :

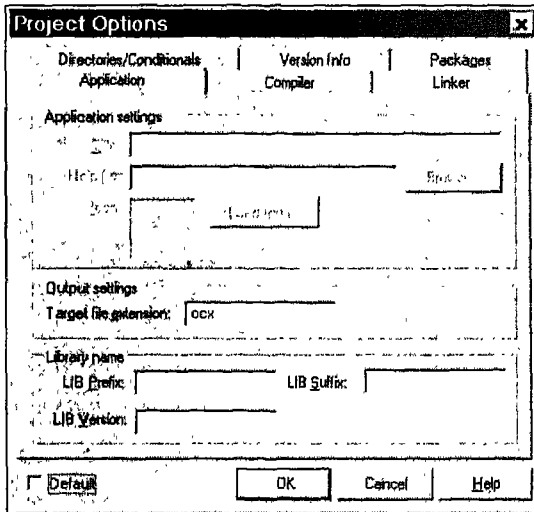
**الأمـر Web Deploy بالقائمة Project**

من خلال الأمر Web Deploy يمكن تهيئة أدوات التحكم ActiveX أو الفورم التى من النوع ActiveForm التى تم الإنتهاء من إعدادها وهذه التهيئة بهدف جعل هذه العناصر متاحة للاستخدام فى خوادم شبكة الويب.

هذا الأمر ينبغي استخدامه بعد استخدام الأمر Web deployment Options وإلا فلن يعمل.

**الأمـر Options بالقائمة Project**

اختيار الأمر Options من القائمة Project يؤدي إلى فتح صندوق الحوار Project Options الموضح فى الشكل التالى :

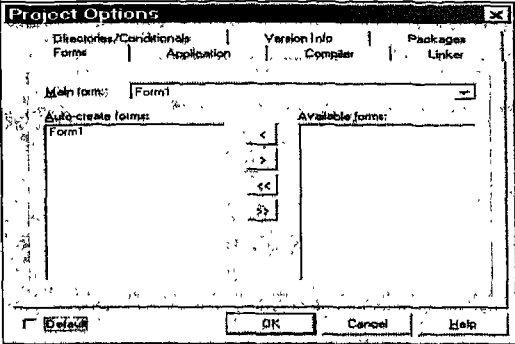
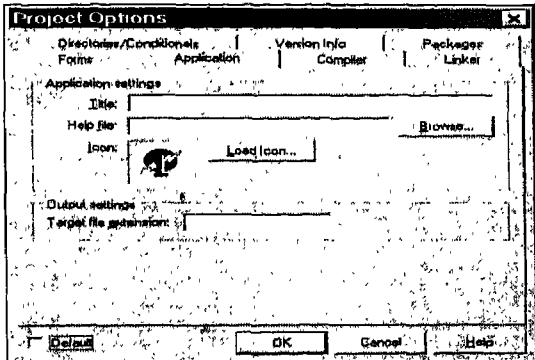


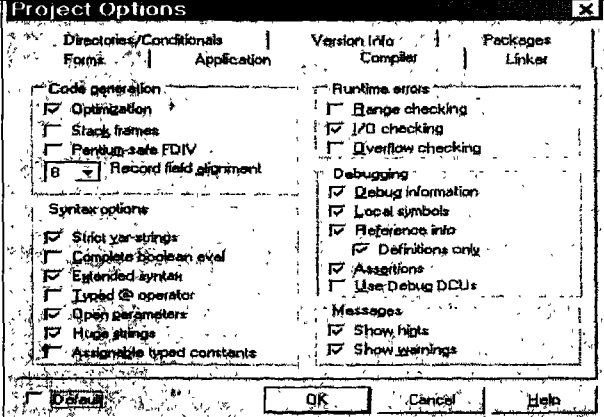
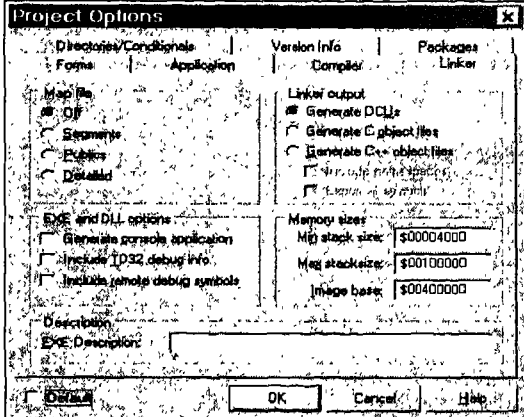
شكل توضيحي :

هناك طريقة أخرى لفتح صندوق الحوار Project Options وهى الضغط على مجموعة المفاتيح Shift+Ctrl+F11 بلوحة المفاتيح.



الجدول التالى يقدم لنا وصف مختصر لوظيفة كل تبويب من التبويبات الموجودة فى صندوق الحوار Project Options :

الوصف والاستخدام	التبويب
<p>التحكم فى الفورم التى يتم إنشاؤها تلقائيا بواسطة المشروع. هذا والشكل التالى يوضح لنا محتويات هذا التبويب :</p> 	Forms
<p>تحديد عنوان title للمشروع واسم ملف المساعدة واسم الأيقونة الملحقة بالمشروع. هذا والشكل التالى يوضح لنا محتويات هذا التبويب :</p> 	Application
<p>تحديد المحولات الخاصة بمرجم اللغة التى تتحكم فى طريقة ترجمة الكود البرمجى. هذا والشكل التالى يوضح لنا محتويات هذا التبويب :</p>	Compiler

	
<p>من خلال هذا التبويب يمكن التحكم فى الطريقة التى يتم بها ربط ملفات المشروع. هذا والشكل التالى يوضح لنا محتويات هذا التبويب :</p> 	Linker
<p>تحديد مواضع الملفات الضرورية لترجمة وربط المشروع الذى يتم العمل به حالياً. هذا والشكل التالى يوضح لنا محتويات هذا التبويب :</p>	Directories/Conditional



Project Options

Formes | Application | Compiler | Linker
Directories/Conditionals | Version Info | Packages

Directories

Output directory: [] ...

Unit output directory: [] ...

Search path: [] ...

Debug source path: [] ...

BPL output directory: [] ...

DCE output directory: [] ...

Conditionals

Conditional defines: [] ...

Aliases

Unit aliases: [WinTypes=Windows,WinProcs=Windows.Db] ...

☒ Default

OK Cancel Help

تحديد نوعية المعلومات التي يتم إلحاقها بالمشروع ومنها معلومات عن رقم الإصدار وغيرها. هذا والشكل التالي يوضح لنا محتويات هذا التبويب :

Version Info

Project Options

Formes | Application | Compiler | Linker
Directories/Conditionals | Version Info | Packages

☒ Include version information in project

Module version number

Major version: [0] Minor version: [0] Patch: [0] Build: [0]

☐ Add version information to build number

Module attributes

☐ Debug build ☐ Symbol build ☐ Force build ☐ Opt

Language

Language ID: [00000000] Compiler build: [00000000]

Key: []

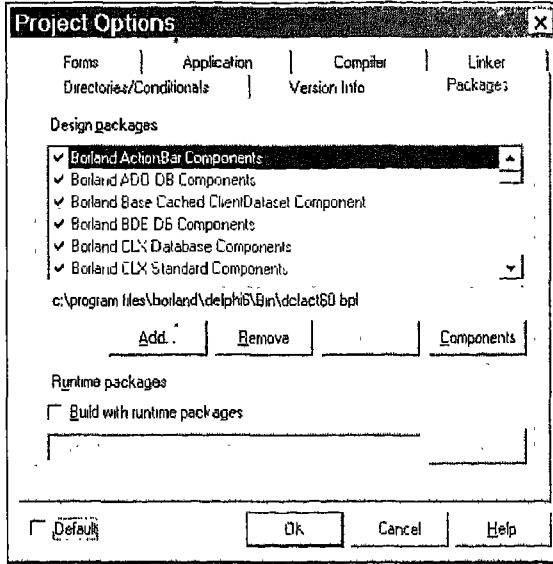
CompanyName: []

☒ Default

OK Cancel Help



تحديد الحزم البرمجية لمرحلة التصميم أو مرحلة التشغيل -Run Time اللازمة لتثبيت التطبيق عند يشرع المستخدمون فى التعامل معه بأجهزتهم. هذا والشكل التالى يوضح لنا محتويات هذا التبويب :



Packages

القائمة Run


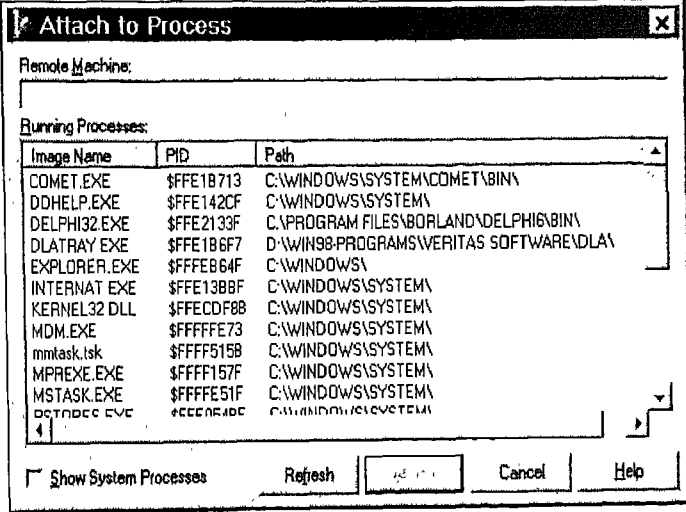

الأوامر الموجودة فى القائمة Run تساعدك فى اكتشاف الثغرات والأخطاء البرمجية الموجودة بالبرنامج الذى تعده. ليس هذا فقط بل تساعدك أيضا فى معالجة هذه الأخطاء والثغرات.

الأوامر الموجودة فى القائمة Run هى المسئولة عن الأداء الوظيفى الجوهري للأداة المتكاملة لفحص الأخطاء ومعالجتها Integrated Debugger.



الجدول التالى يقدم لنا وصف مختصر للأوامر الموجودة فى القائمة Run :



الوصف والاستخدام	الأمر
ترجمة وتنفيذ التطبيق الذي تعده. هذا ويمكن أيضا الضغط على المفتاح F9 بلوحة المفاتيح أو النقر بالماوس على الأيقونة  .	Run
يعمل هذا الأمر على تقديم قائمة بالعمليات التي يتم تشغيلها حاليا والتي يمكنك فحصها لاكتشاف ما بها من أخطاء والعمل على معالجتها وذلك من خلال صندوق الحوار Attach to process الموضح في الشكل التالي :	Attach to Process
	
هذا ويمكن أيضا استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة  .	
يعمل هذا الأمر على توصيف معاملات بداية التشغيل للتطبيق الذي تتولى إعداده كما يعمل على تحديد مضيف قابل للتنفيذ بالنسبة للملفات مكتبات الربط الديناميكي DLLs وذلك من خلال التبويب Local بصندوق الحوار Run Parameters الموضح في الشكل التالي :	Parameters



The 'Run Parameters' dialog box is shown with the 'Local' tab selected. It contains a 'Host Application' dropdown menu with a 'Browse...' button next to it. Below that is a 'Parameters' dropdown menu. At the bottom are buttons for 'OK', 'Cancel', and 'Help'.

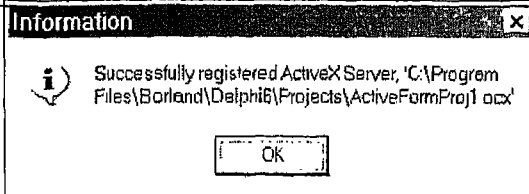
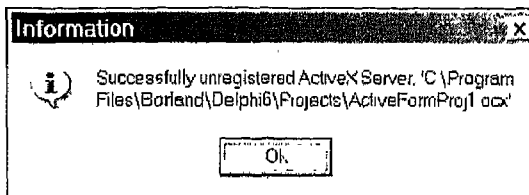
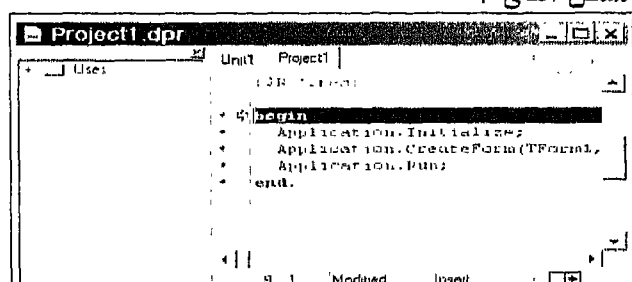
كما يمكن أيضا إجراء نفس ما سبق في كمبيوتر من أجل إجراء عملية اكتشاف الأخطاء معالجتها من على بعد عن طريق الشبكات وذلك عن طريق التبويب Remote بصندوق الحوار Run Parameters الموضح في الشكل التالي :

The 'Run Parameters' dialog box is shown with the 'Remote' tab selected. It contains a 'Remote Path' dropdown menu, a 'Remote Host' dropdown menu, and a 'Parameters' dropdown menu. At the bottom, there is a checkbox labeled 'Debug project on remote machine' and buttons for 'OK', 'Cancel', and 'Help'.

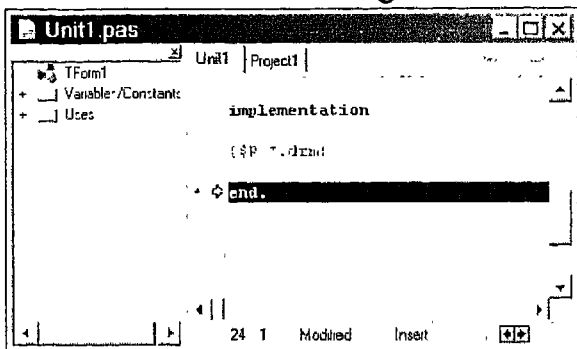
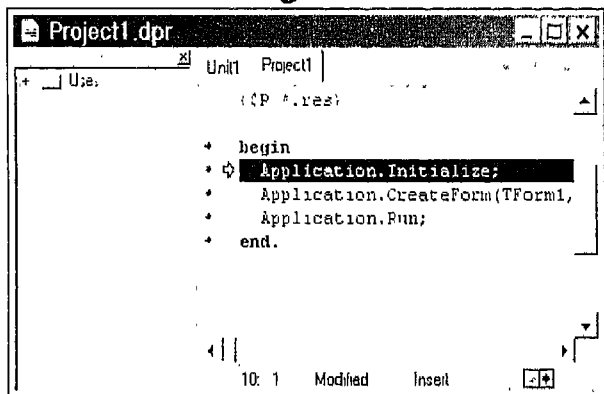
يعمل هذا الأمر على إضافة مدخل أحقية الاستخدام لبيئة الويندوز لأدوات التحكم ActiveX التي تعدها وهذا الأمر يكون متاح للاستخدام عندما يكون المشروع الحالي عبارة عن ActiveX Project. هذا وبعد الإنتهاء من استخدام هذا الأمر تظهر على الشاشة الرسالة الموضحة في الشكل التالي :

Register ActiveX
Server



	
<p>يعمل هذا الأمر على إزالة المشروع من أحقية استخدام بيئة الويندوز. وهذا الأمر يكون متاح للاستخدام عندما يكون المشروع الحالى عبارة عن ActiveX Project. هذا وبعد الإنتهاء من استخدام هذا الأمر تظهر على الشاشة الرسالة الموضحة فى الشكل التالى :</p> 	<p>Unregister ActiveX Server</p>
<p>من خلال هذا الأمر يتم تركيب وتثبيت كائنات فى المشروع الحالى داخل تطبيق من النوعية COM+. ويظهر هذا الأمر فى القائمة Run عندما يكون نظام التشغيل مدعما للتكنولوجيا COM+.</p>	<p>Install COM+ Objects</p>
<p>من خلال هذا الأمر يتم تنفيذ الكود البرمجى سطر بسطر وفى أثناء ذلك يتم الإشارة إلى السطر الذى يتم تنفيذه حاليا بسهم أخضر فى نافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :</p> 	<p>Step Over</p>

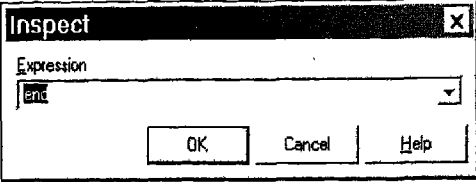
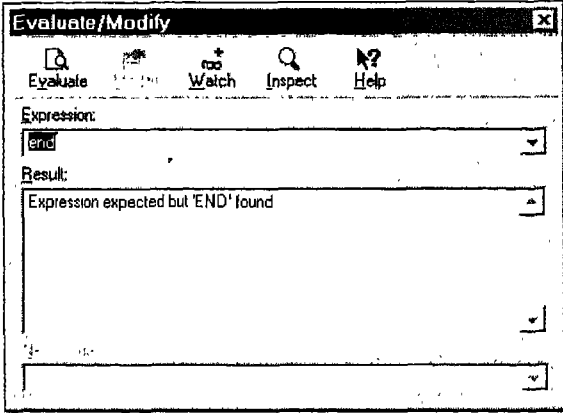


<p>وهذه العملية تتم عبر الإجراءات في أثناء تنفيذهم كما لو كانوا وحدة واحدة. ونود هنا القول بأنه يمكن استدعاء هذا الأمر عن طريق الضغط على المفتاح F8 بلوحة المفاتيح.</p>	
<p>يعمل هذا الأمر على تنفيذ الكود البرمجي سطر بسطر مع إجراء عملية تتبع داخل الإجراءات بنافذة محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالي :</p>  <p>هذا ويمكن استدعاء الأمر عن طريق الضغط على المفتاح F7 بلوحة المفاتيح.</p>	Trace Into
<p>يعمل هذا الأمر يقوم بتنفيذ الكود البرمجي وفي أثناء ذلك يتم التوقف عند السطر التالي القابل للتنفيذ بنافذة محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالي :</p> 	Trace To Next Source Line

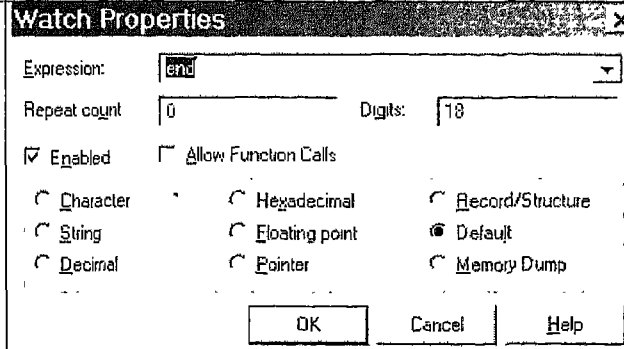


<p>هذا ويمكن استدعاء الأمر عن طريق الضغط على المفاتيح Shift+F7 بلوحة المفاتيح.</p>	
<p>يقوم هذا الأمر بتشغيل البرنامج المحمل حتى يتم الوصول إلى موضع مؤشر الكتابة فى الكود البرمجى داخل نافذة محرر الكود البرمجى Code Editor. هذا ويمكن استدعاء الأمر عن طريق الضغط على المفتاح F4 بلوحة المفاتيح.</p>	<p>Run To Cursor</p>
<p>يقوم هذا الأمر بتشغيل البرنامج حتى يتم الحصول على أى مخرجات من الدالة التى يتم التعامل معها حالياً. هذا ويمكن استدعاء الأمر عن طريق الضغط على المفاتيح Shift+F8 بلوحة المفاتيح.</p>	<p>Run Until Return</p>
<p>يعمل هذا الأمر على وضع مؤشر الكتابة عند نقطة التنفيذ فى النافذة CPU كما هو موضح بالشكل التالى :</p> 	<p>Show Execution Point</p>
<p>من خلال هذا الأمر يمكن تعليق تنفيذ البرنامج بشكل مؤقت.</p>	<p>Program Pause</p>
<p>يقوم هذا الأمر بإنهاء تشغيل التطبيق وإزالته من ذاكرة الكمبيوتر العشوائية وجعل التطبيق ينتقل من مرحلة التشغيل Run-Time إلى مود التصميم. هذا ويمكن استدعاء الأمر عن طريق الضغط على المفاتيح Ctrl+F2 بلوحة المفاتيح.</p>	<p>Program Reset</p>



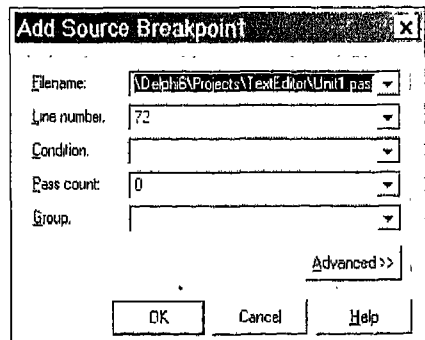
<p>يعمل هذا الأمر على فتح صندوق الحوار Inspector الموضح فى الشكل التالى :</p> 	Inspect
<p>يعمل هذا الأمر على عرض النافذة Evaluate/Modify الموضحة فى الشكل التالى :</p> 	Evaluate/Modify
<p>يعمل هذا الأمر على فتح صندوق حوار خصائص المراقبة Watch Properties الموضح فى الشكل التالى :</p>	Add Watch



 <p>بصندوق الحوار هذا تستطيع أن إنشاء وتعديل أدوات المراقبة التي يتم استخدامها في أثناء متابعة تنفيذ الكود البرمجي. هذا ويمكن استدعاء الأمر عن طريق الضغط على المفاتيح Ctrl+F5 بلوحة المفاتيح.</p>	
<p>لهذا الأمر قائمة فرعية تشتمل على مجموعة من الأوامر التي يمكن من خلالها إنشاء وتعديل نقط التوقف بالكود البرمجي أثناء تنفيذه في مرحلة التشغيل Run-Time.</p>	<p>Add Breakpoint</p>

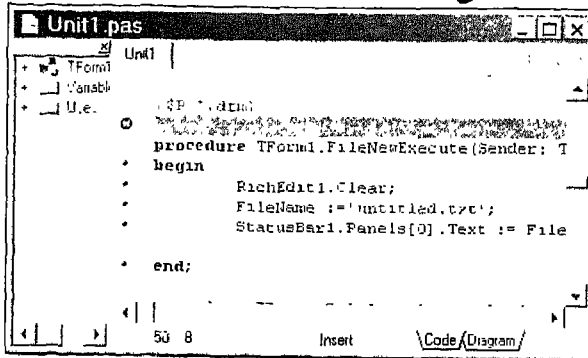
الجدول التالي يقدم لنا وصف مختصر لوظيفة كل أمر من الأوامر الموجودة بالقائمة


الفرعية الخاصة بالأمر Add Breakpoint :

الوصف والاستخدام	الأمر
<p>يعمل هذا الأمر على فتح صندوق الحوار Add Source Breakpoint الموضح في الشكل التالي :</p> 	<p>Source Breakpoint</p>

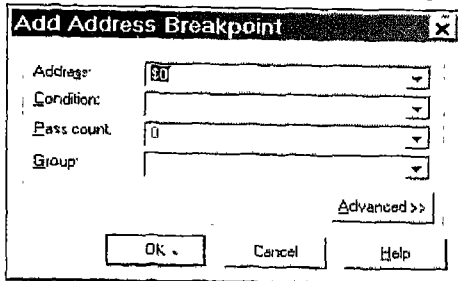


بصندوق الحوار هذا تستطيع أن تضع نقطة توقف في سطر معين بالكود البرمجي المصدري بالتطبيق الذى تعمل به. ونتيجة ذلك أنه عندما تقوم بتشغيل التطبيق تجد أن نقطة التنفيذ في محرر الكود البرمجي Code Editor تشير إلى موضع نقطة التوقف كما هو موضح بالشكل التالى :



هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .



يعمل هذا الأمر على فتح صندوق الحوار Add Address Breakpoint الموضح فى الشكل التالى :

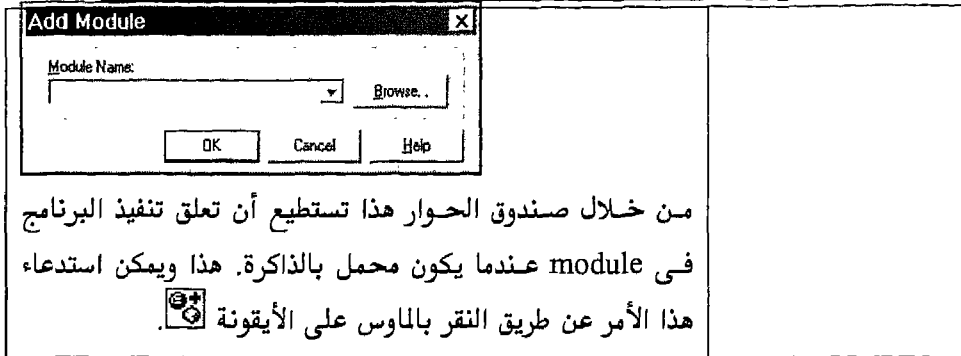


بصندوق الحوار هذا تستطيع أن تضع نقطة توقف فى آلية معينة من آليات تنفيذ الأوامر machine instruction. وعندما تقوم بتشغيل التطبيق نجد أن نقطة التنفيذ فى النافذة CPU تشير إلى موضع نقطة التوقف كما هو موضح بالشكل التالى :

Address
Breakpoint

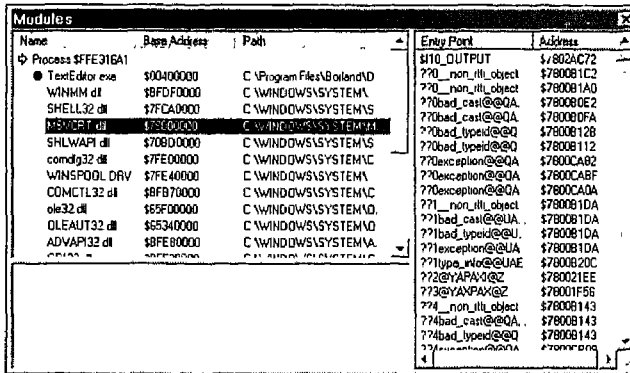


<p>هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .</p>	
<p>عن طريق هذا الأمر يتم فتح صندوق الحوار Add Data Breakpoint Breakpoint الموضح فى الشكل التالى :</p> <p>بصندوق الحوار هذا تستطيع وضع نقطة توقف فى عنوان معين لكى تعمل على إيقاف تنفيذ البرنامج عندما تصل نقطة التنفيذ إلى الموضع الموجودة فيه نقطة التوقف. هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .</p>	<p>Data Breakpoint</p>
<p>عن طريق هذا الأمر يتم فتح صندوق الحوار Add/Edit Module Breakpoint الموضح فى الشكل التالى :</p>	<p>Module Load Breakpoint</p>



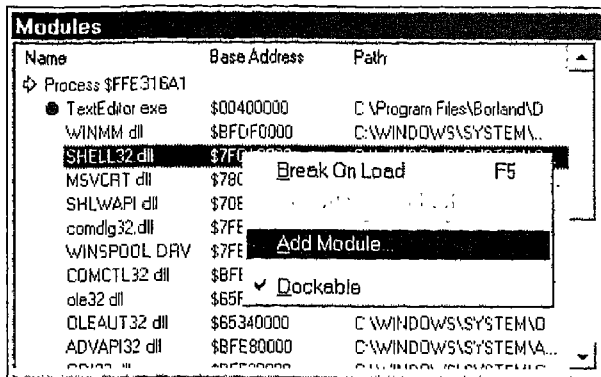
خطوات إضافة نقط توقف من خلال مشاهد اكتشاف ومعالجة الأخطاء debugging views

- (١) قم أولاً بتشغيل التطبيق الذى تعده. ثم افتح القائمة View واختر منها الأمر Debug Windows ثم اختر Modules لتظهر على الشاشة النافذة Module List الموضحة فى الشكل التالى :



شكل توضيحي :

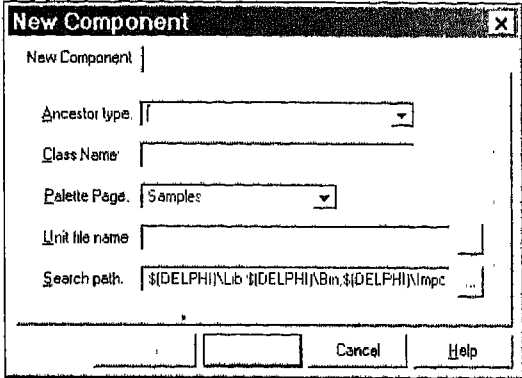

- (٢) انقر بالزر الأيمن على أى Module من Modules الموجودة فى الجانب الأيسر العلوى من النافذة Modules List ثم اختر Add module من القائمة المختصرة التى تظهر على الشاشة كما هو موضح بالشكل التالى :



شكل توضيحي :

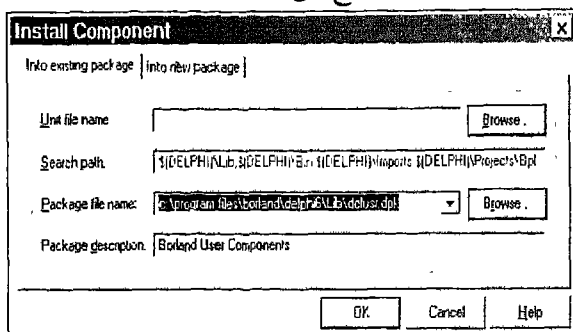
القائمة Component

تشتمل القائمة Component مجموعة الأوامر التالية :

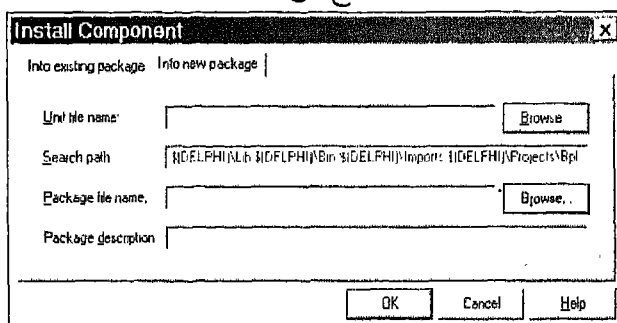
الوصف والاستخدام	الأمر
<p>يقوم هذا الأمر بفتح صندوق الحوار New Component الموضح فى الشكل التالى :</p>  <p>بصندوق الحوار هذا تستطيع إنشاء مكون جديد من خلال تحديد كل من نوع الكائن الذى سينحدر منه واسم القطاع الذى سيوجد به واختيار الباليتة التى سيوضع بها بباليتة المكونات وتحديد اسم ملف الوحدة الخاص به وتحديد أين سيتم تخزينه. هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .</p>	New Component



يقوم هذا الأمر بتركيب وتثبيت مكون داخل حزمة برمجية موجودة بالفعل وذلك من خلال التبويب Into Existing Package بصندوق الحوار Install Component الموضح في الشكل التالي :



كما يمكن أيضا تركيب وتثبيت مكون داخل حزمة برمجية جديدة بالفعل وذلك من خلال التبويب Into New Package بصندوق الحوار Install Component الموضح في الشكل التالي :



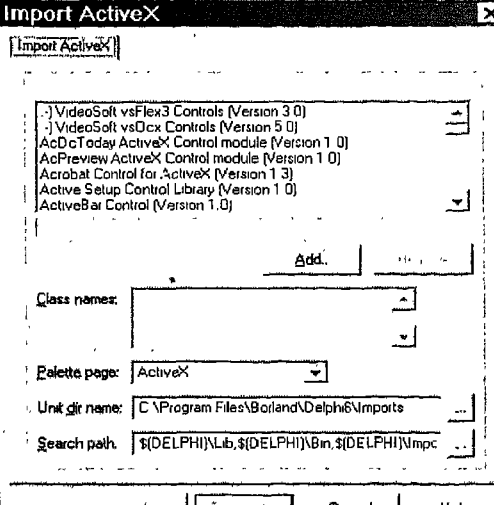

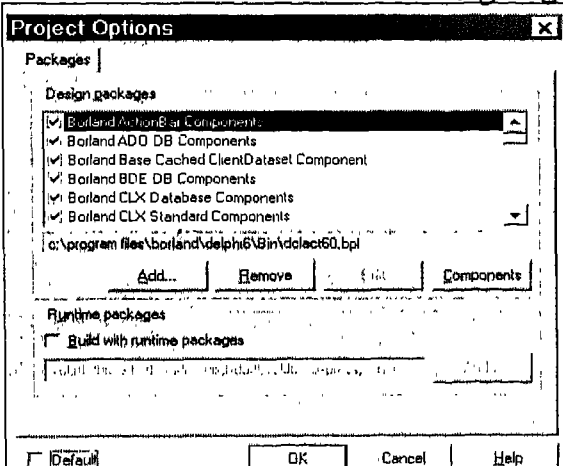
هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة




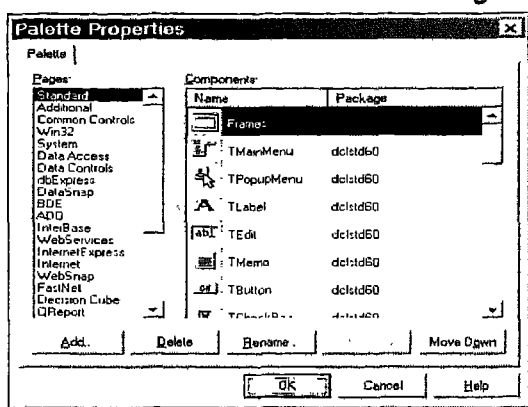
يعمل هذا الأمر على إضافة مكتبات الأنواع الخاصة بأدوات التحكم إلى المشروع الذي تعده وهذه الإضافة تتم عن طريق صندوق الحوار Import ActiveX الموضح في الشكل التالي :

Install
Component

Import
ActiveX
Control

	<p>هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة </p>	
<p>من خلال هذا الأمر يمكن تفصيل مكونات وحفظها على أساس كونها قوالب بأسماء جديدة وفي صفحة جديدة بالليته المكونات وبأيقونات خاصة بهم أيضا.</p>		<p>Create Component Template</p>
<p>من خلال هذا الأمر يتم فتح صندوق الحوار Project Options الموضح في الشكل التالي :</p> 	<p>Install Packages</p>	

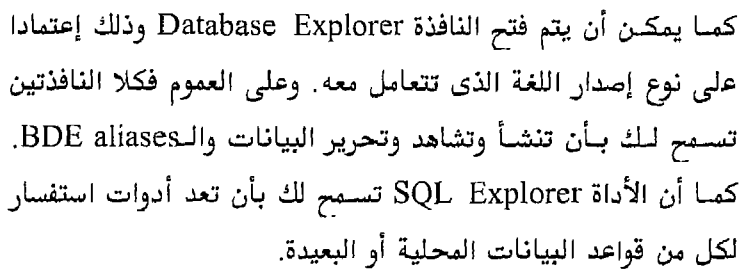


<p>بصندوق الحوار هذا يتم تحديد الحزم البرمجية التي يحتاج إليها المشروع الذي تعدده. هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .</p>	
<p>يعمل هذا الأمر على فتح صندوق الحوار Palette Properties الموضح في الشكل التالي :</p>  <p>من خلال صندوق الحوار هذا يمكن إضافة مكونات جديدة إلى باليتة المكونات كما يمكن أيضا إزالة أي مكونات منها وكذلك تغيير اسماء أي من المكونات الموجودة بها بالإضافة إلى إمكانية تغيير ترتيب أيقونات المكونات في كل صفحة من صفحات باليتة المكونات.</p>	<p>Configure Palette</p>

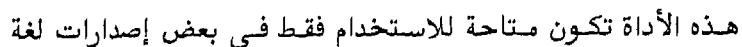
القائمة Database

الأوامر الموجودة في القائمة Database تمكّنك من إنشاء وتعديل وتتبع ومشاهدة قواعد البيانات. هذا والجدول التالي يقدم لنا وصفا مختصرا لوظيفة الأوامر الموجودة بهذه القائمة :

الوصف والاستخدام	الأمر
<p>يعمل هذا الأمر على فتح النافذة SQL Explorer الموضحة في الشكل التالي :</p>	<p>Explorer</p>



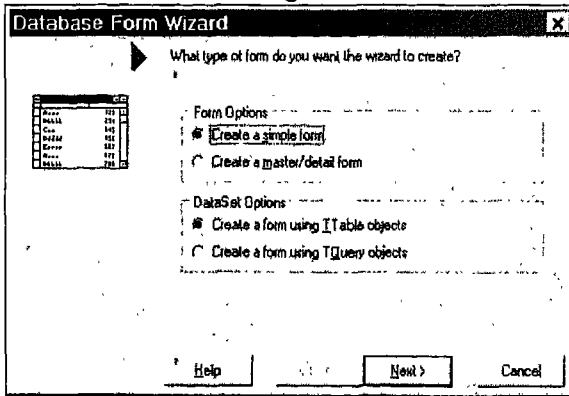
SQL Monitor



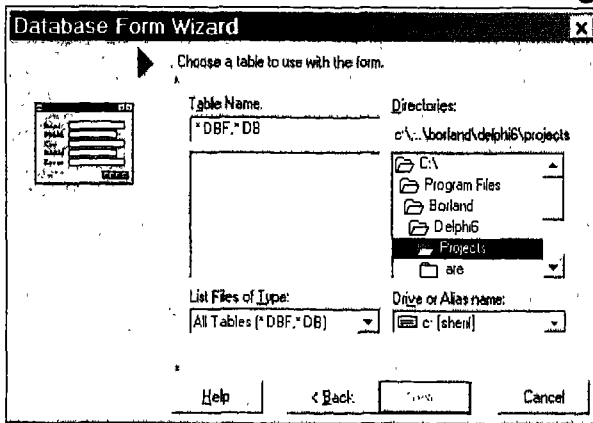
Delphi وهي تسمح لك بأن تراقب SQL resource allocation ومشاهدة جمل الاستدعاءات الحقيقية التي يتم التعامل معها من خلال أدوات الربط SQL Links وهذه الإستدعاءات تكون لخادم بعيد أو لمصدر من مصادر البيانات ODBC وذلك من خلال ODBC Socket.

Form Wizard

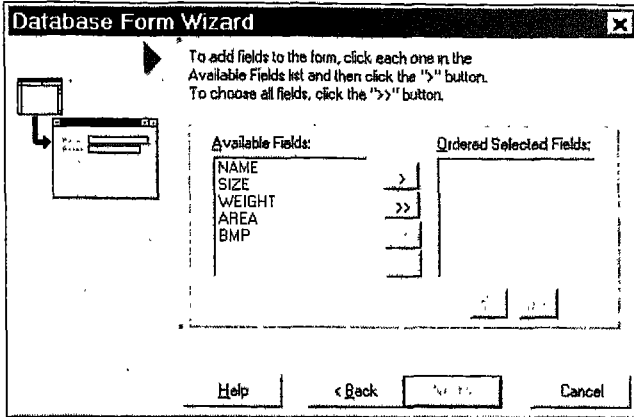
عند اختيار هذا الأمر يتم فتح أول صندوق حوار فى المعالج Database Form Wizard (المستخدم لإنشاء فورمة تعرض بيانات من قواعد بيانات محلية أو بعيدة) الموضح فى الشكل التالى :



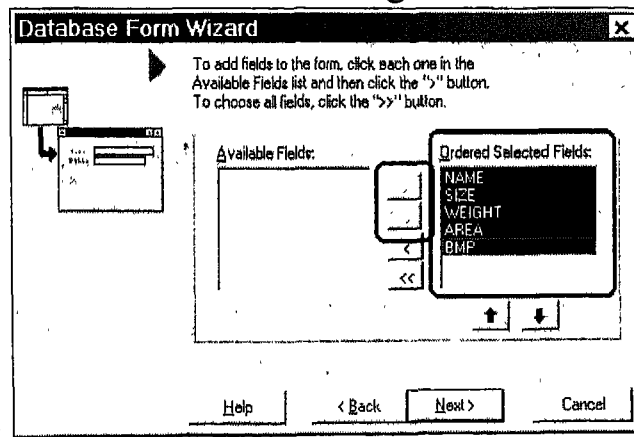
صندوق الحوار هذا يطلب تحديد نوعية الفورمة التى تود إنشاؤها. بعد ذلك انقر بالماوس على المفتاح Next للانتقال إلى صندوق الحوار التالى والموضح بالشكل التالى :



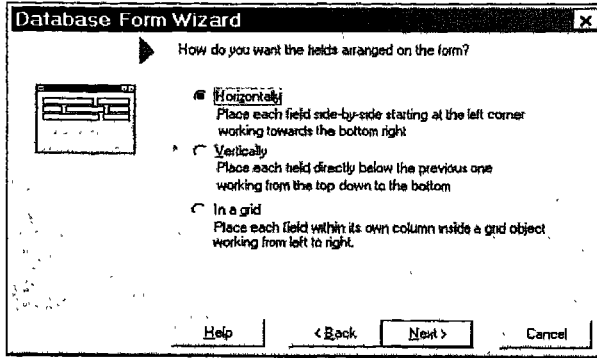
بصندوق الحوار هذا تستطيع اختيار الجدول الذى سيكون مصدر للبيانات التى سيتم عرضها فى الفورمة. بعد ذلك انقر بالماوس على المفتاح Next للانتقال إلى صندوق الحوار التالى والموضح بالشكل التالى:



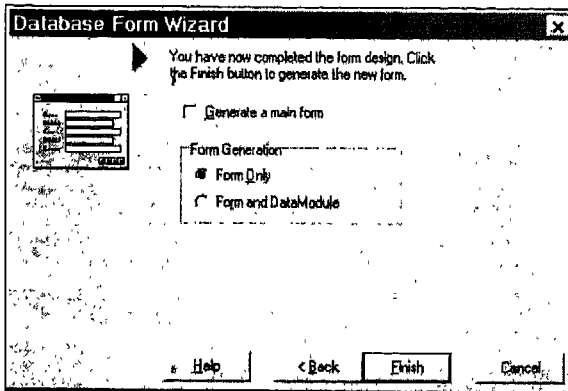
بصندوق الحوار هذا تستطيع اختيار الحقول التى سيتم وضعها فى الفورمة وللقيام بذلك علم بالماوس على الحقول التى ترغبها فى قائمة العرض Available Fields ثم انقر بالماوس على الأيقونة > (أو انقر بالماوس على الأيقونة >> لنقل الحقول كلها من قائمة العرض Available Fields إلى قائمة العرض Ordered Selected Fields كما هو موضح بالشكل التالى :



والآن انقر بالماوس على المفتاح Next للانتقال إلى صندوق الحوار التالى والموضح بالشكل التالى :



صندوق الحوار هذا يطلب منك تحديد طريقة ترتيب الحقول المختارة فى الفورمة (أفقيا أو رأسيا أو فى مجموعة خلايا كما فى برامج الجداول الإلكترونية مثل برنامج Excel). بعد ذلك انقر بالماوس على المفتاح Next للانتقال إلى صندوق الحوار التالى والموضح بالشكل التالى :



يعتبر صندوق الحوار هذا آخر صندوق حوار فى المعالج Database Form Wizard وهو يطلب منك أن تنقر بالماوس على المفتاح Finish للبدء فى تكوين الفورمة والتى يمكن أن تكون مشابهة للفورمة الموضحة فى الشكل التالى :



القائمة Tools

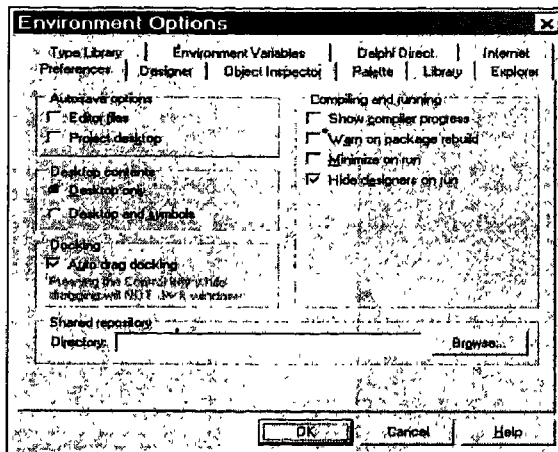
- يمكن استخدام قائمة الأدوات Tools للقيام بالآتي :
- مشاهدة القيم التحديدية التي تتحكم في خصائص بيئة العمل مع إمكانية تغيير أى من هذه القيم التحديدية.
- مشاهدة القيم التحديدية التي تتحكم في خصائص عملية التحرير Editing مع إمكانية تغيير أى من هذه القيم التحديدية.
- مشاهدة القيم التحديدية التي تتحكم في خصائص وصفات أداة اكتشاف ومعالجة الأخطاء والثغرات Debugger مع إمكانية تغيير أى من هذه القيم التحديدية.
- التعديل في العناصر الموجودة في مستودع الكائنات Object Repository.
- التعديل في مجموعة البرامج الموجودة في القائمة Tools.
- إنشاء وتعديل جداول قواعد البيانات المحلية.
- إنشاء وتعديل مجموعات الحزم البرمجية.
- إنشاء وتحرير الصور.
- فيما يلي سنستعرض سويا من خلال الجدول التالي الأوامر الأساسية Default الموجودة في القائمة Tools :

الوصف والاستخدام

الأمر

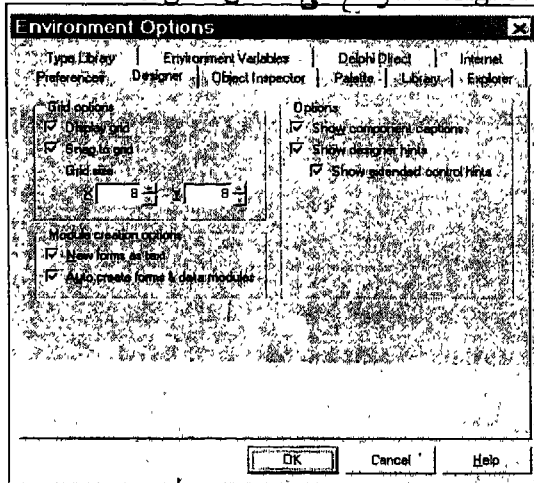
Environment Options

من خلال هذا الأمر يتم عرض صندوق حوار خيارات بيئة العمل Environment Options الموضح في الشكل التالي :



عن طريق صندوق الحوار هذا تستطيع أن تقوم بكل من الآتي :

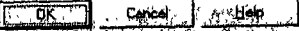
- تحديد القيم المفضلة لتهيئة بيئة العمل وذلك من خلال التبويب Preferences الموضح في الشكل السابق.
- تحديد القيم المفضلة لأداة تصميم الفورم وذلك من خلال التبويب Designer الموضح في الشكل التالي :



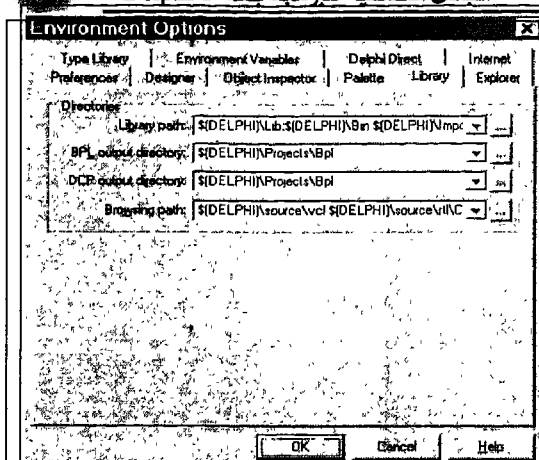
التالى :



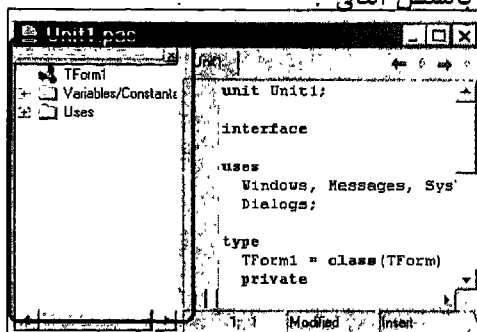
التحكم



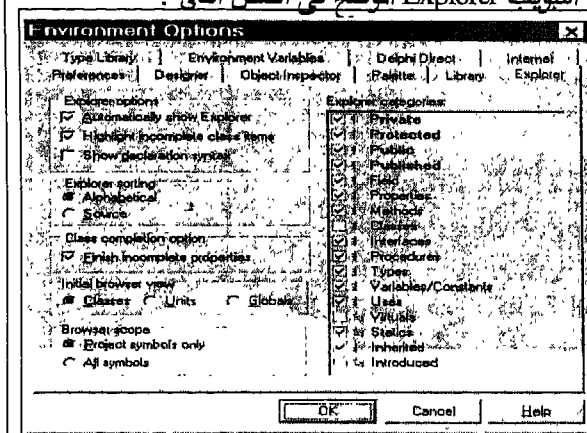
تحکم ف



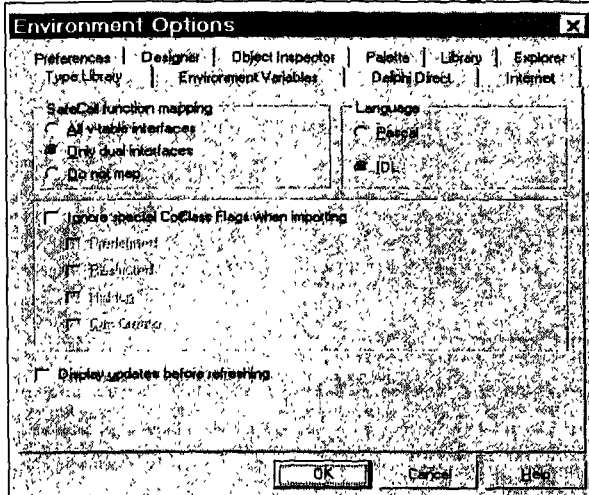
التحكم في خصائص متصفح الكود البرمجي الموجود في الجانب الأيسر بنافذة محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالي :



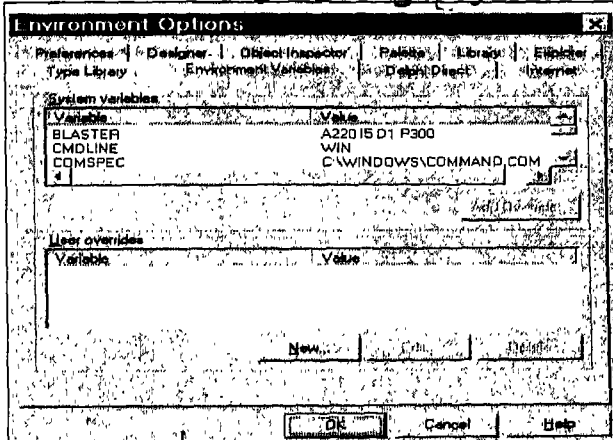
وذلك من خلال التبويب Explorer الموضح في الشكل التالي :



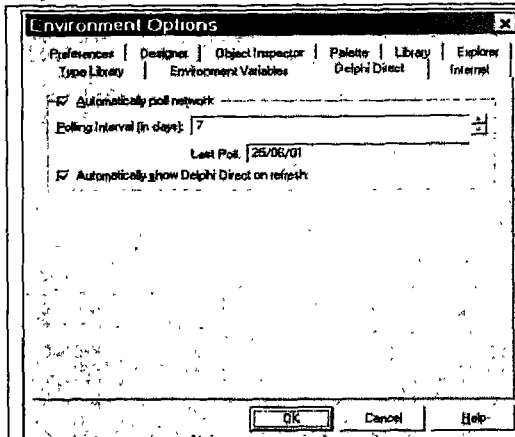
● التحكم فى خصائص مكتبات الأنواع والمسارات الدالة على وجودها وذلك من خلال التبويب Type Library الموضح فى الشكل التالى :



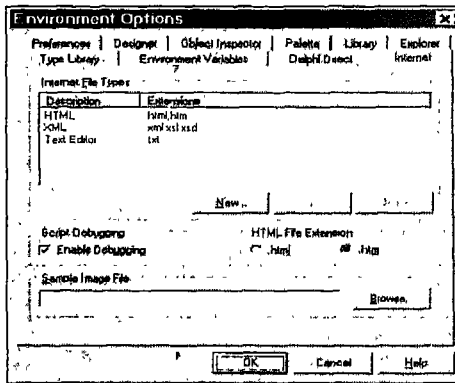
● التحكم فى القيم الخاصة بالمتغيرات التى تحدد صفات بيئة العمل وذلك من خلال التبويب Environment Variables الموضح فى الشكل التالى :



● التحكم فى إمكانية الوصول إلى المصادر المتاحة بشبكة الإنترنت للغة Delphi وذلك من خلال التبويب Delphi Direct الموضح فى الشكل التالى :



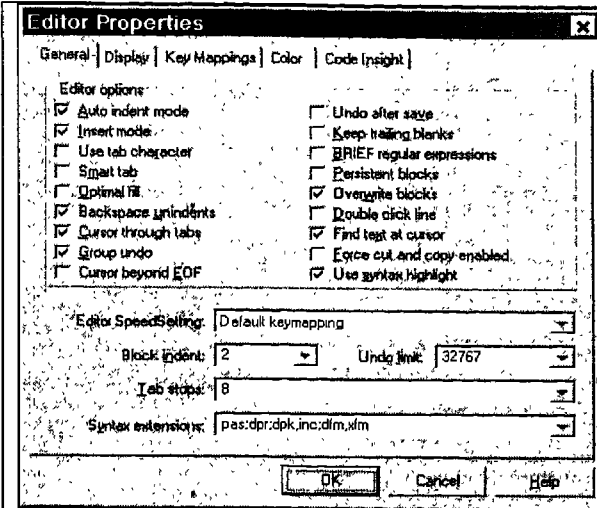
● التحكم في خصائص إمكانية تفاعل لغة Delphi مع الملفات المعدة بلغات البرمجة للإنترنت مثل لغة XML ولغة HTML وذلك من خلال التبويب Internet الموضح في الشكل التالي :



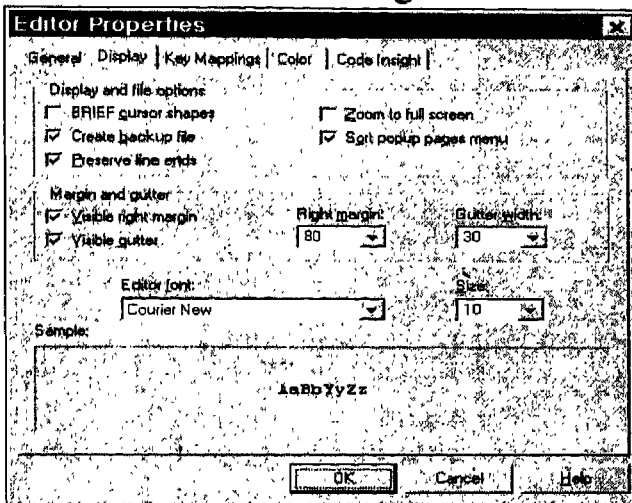
يعمل هذا الأمر على فتح صندوق الحوار Editor Properties الذي يمكن من خلاله تحديد القيم المفضلة لتهيئة محرر الكود البرمجي Code Editor. وهذه التهيئة تتم من خلال الآتي :

التحكم في الخصائص العامة لمحرر الكود البرمجي وذلك عن طريق التبويب General الموضح في الشكل التالي :

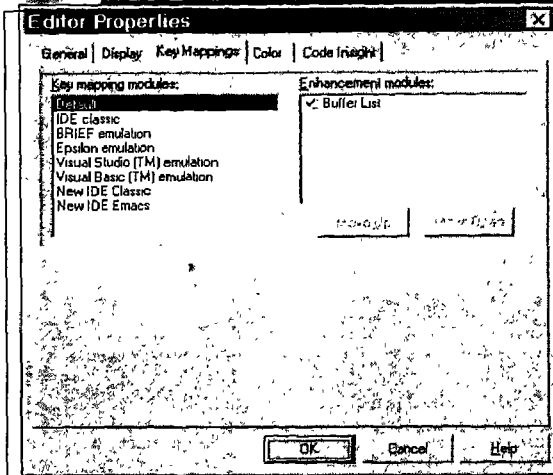
Editor Options



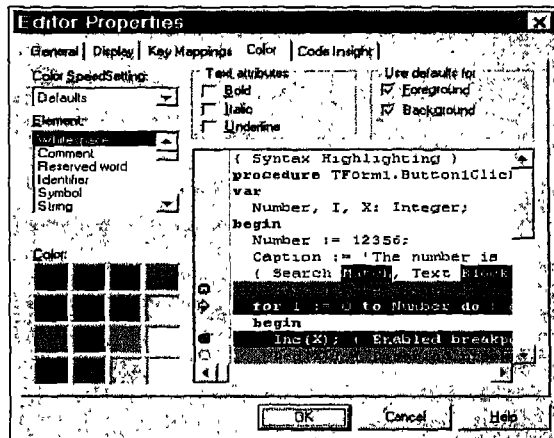
التحكم فى طريقة عرض النصوص بمحرر الكود البرمجى وذلك عن طريق التبويب Display الموضح فى الشكل التالى :



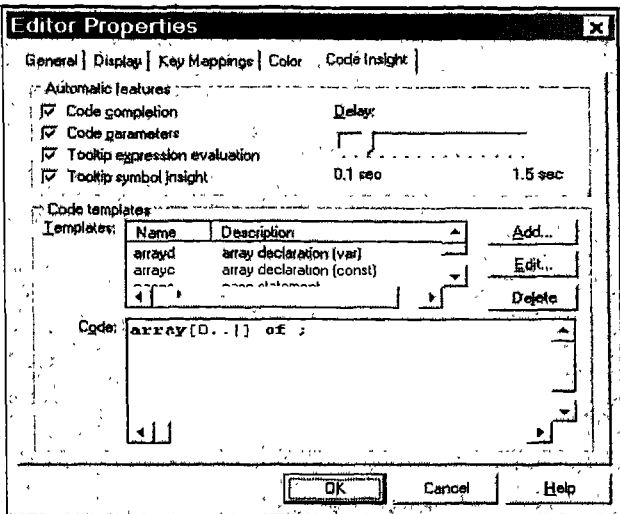
التحكم فى خصائص Modules key mapping مع إمكانية تمكين أو تعطيل أى منها بمحرر الكود البرمجى وذلك عن طريق التبويب Key Mapping الموضح فى الشكل التالى :



التحكم في طريقة تلوين أجزاء معينة في الكود البرمجي مثل الكلمات المحجوزة وأسماء الدوال... وذلك عن طريق التبويب Color الموضح في الشكل التالي :

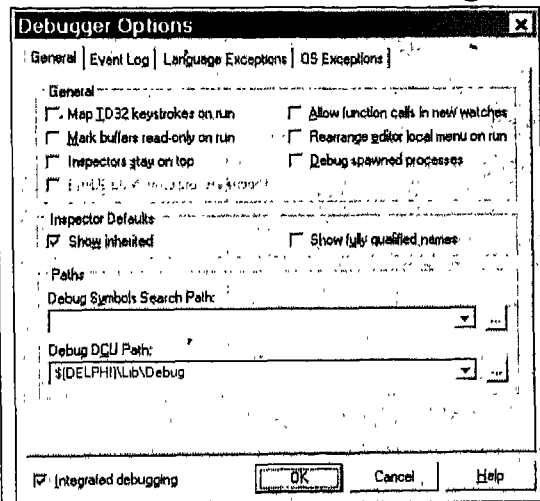


التحكم في طريقة مشاهدة الكود البرمجي بمحرر الكود البرمجي وفي إمكانية الاستكمال التلقائي للجمل في أثناء كتابتها وذلك عن طريق التبويب Code Insight الموضح في الشكل التالي :



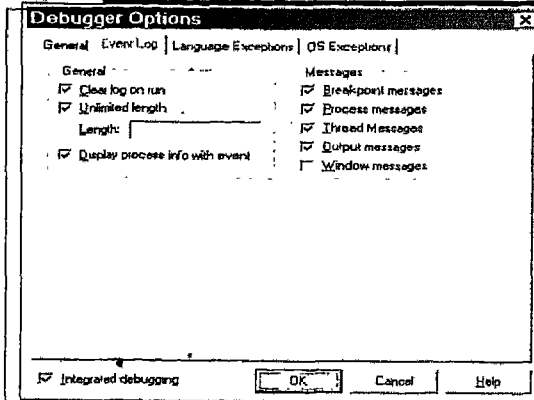
يعمل هذا الأمر على فتح صندوق الحوار Debugger Options والذي يمكن من خلاله التحكم في خصائص وصفات أداة اكتشاف ومعالجة الأخطاء والثغرات وهذا التحكم يتم من خلال التبويبات الأربعة التالية :

التبويب General الموضح في الشكل التالي :

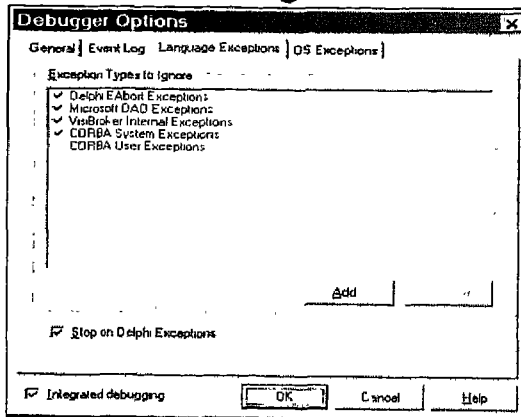


التبويب Event Log الموضح في الشكل التالي :

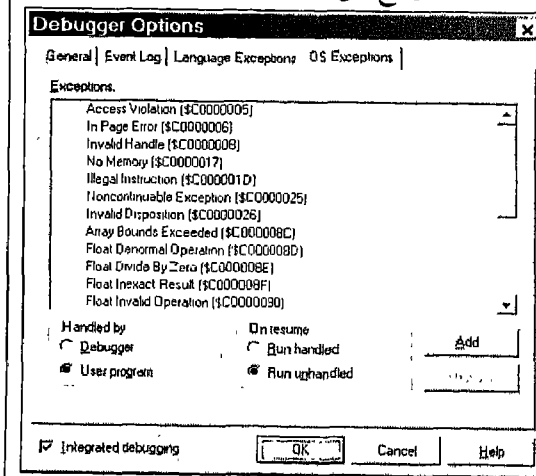
Debugger Options



التبويب Language Exceptions الموضح في الشكل التالي :

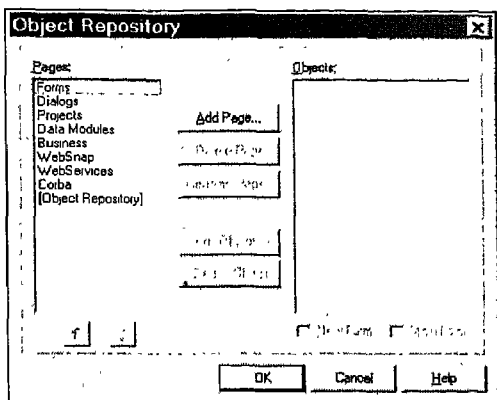


التبويب OS Exceptions الموضح في الشكل التالي :





يعمل هذا الأمر على فتح صندوق الحوار Object Repository
الموضح فى الشكل التالى :



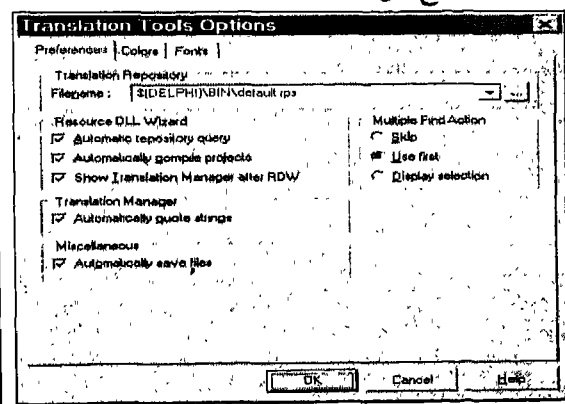
Repository

من خلال صندوق الحوار هذا يمكن إضافة صفحات جديدة إلى
مستودع الكائنات كما يمكن أيضا مسح أى صفحة من الصفحات
الموجودة بالفعل كما نستطيع أيضا تغيير اسم أى من هذه الصفحات

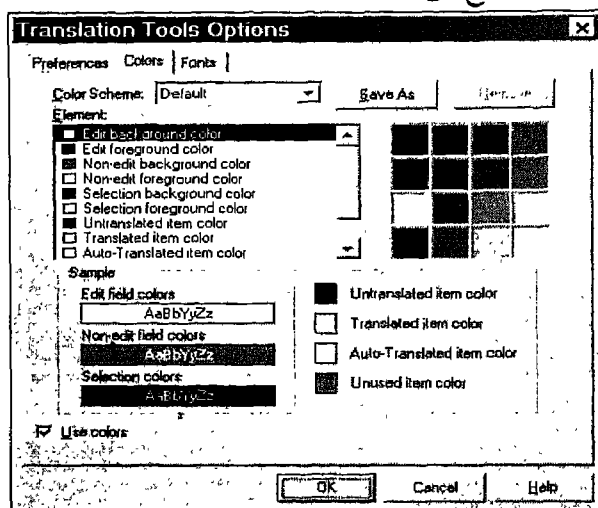
عن طريق هذا الأمر يتم فتح صندوق الحوار Translation Tools
Options والذي يمكن من خلاله تهيئة وإعداد أدوات الترجمة
للغوية Translation Tools وذلك من خلال التبويبات الثلاثة
الموجودة به :

Translation
Tools Options

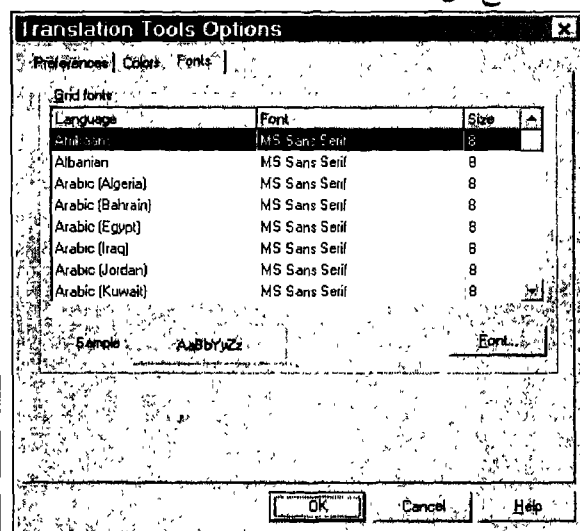
التبويب Preferences الموضح فى الشكل التالى :



التبويب Colors الموضح في الشكل التالي :



التبويب Fonts الموضح في الشكل التالي :



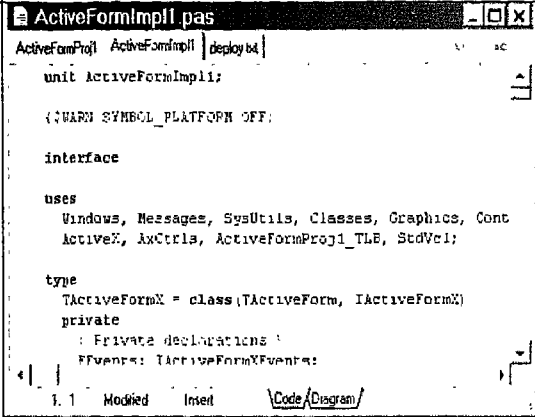
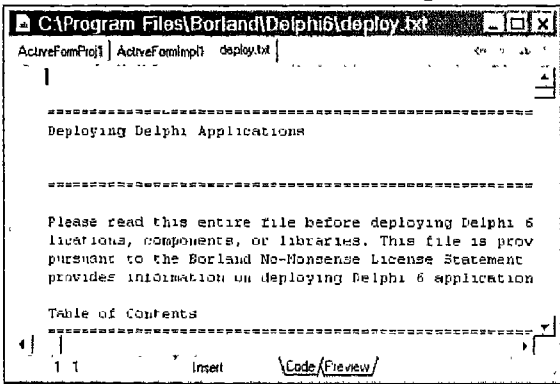
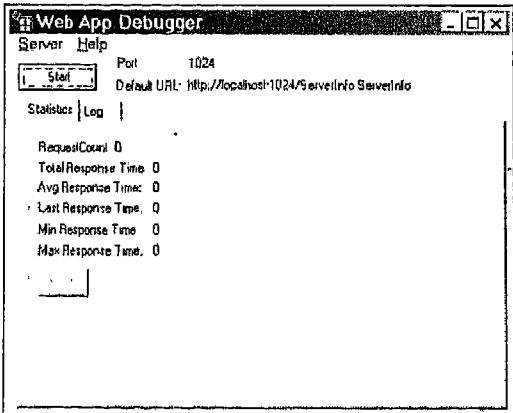
يعمل هذا الأمر على فتح نافذة مستودع الترجمة Translation Repository الموضح في الشكل التالي :

Translation Repository



<p>من خلال هذه النافذة يمكن تخزين واستعادة السلاسل الحرفية المترجمة في مدير الترجمة.</p>	
<p>عن طريق هذا الأمر يتم الولوج إلى محرر صفحات الويب وهذا المحرر ينتمي إلى الطائفة third-party وهذا المحرر يتألف من التبويبات الثلاثة التالية :</p> <p>التبويب ActiveFormProj1 الموضح في الشكل التالي :</p>	<p>External Editor</p>
	<p>التبويب ActiveFormImpl1 الموضح في الشكل التالي :</p>

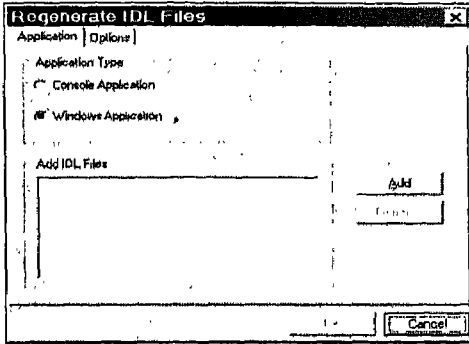


		
التبويب Deploy.txt في الشكل التالي :		
		
يعمل هذا الأمر على فتح النافذة Web App Debugger في الشكل التالي :		
		Web App Debugger

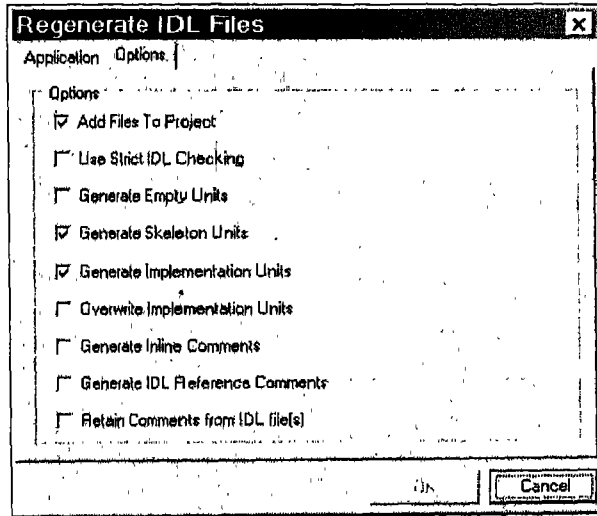
عن طريق هذه النافذة تستطيع أن تختبر تطبيقات التى تعدها للعمل فى خوادم شبكة الويب كما تستطيع أيضا اكتشاف ومعالجة الأخطاء والثغرات التى تظهر بها.

عن طريق هذا الأمر يتم فتح صندوق الحوار Regenerate IDL Files والذى يمكن من خلاله تكوين تطبيق client أو Server مبنى على الملفات التى من النوعية IDL Files. هذا ويتألف صندوق الحوار هذا من التبويبات التالية :

التبويب Application الموضح فى الشكل التالى :



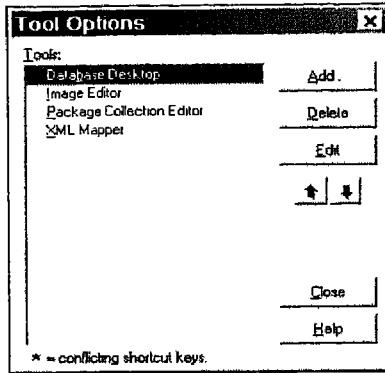
التبويب Options الموضح فى الشكل التالى :



Regenerate
CORBA IDL
Files



يعمل هذا الأمر على فتح صندوق الحوار Tool Options الموضح فى الشكل التالى :

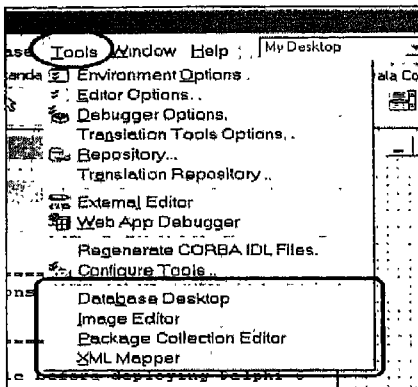


يمكن استخدام صندوق الحوار هذا لإضافة أو إزالة أو تعديل الأوامر الموجودة فى القائمة Tools.

Configure
Tools

الخيارات الإضافية بالقائمة Tools

الجزء السفلى بالقائمة Tools والموضح فى الشكل التالى يمكن تفصيله كما يروق لك :



شكل توضيحي :

فأنت تستطيع إزالة الأدوات الموجودة فى هذا الجزء كما تستطيع أيضا إضافة أدوات أخرى من الأدوات التى سبق تثبيتها أو تفصيلها والتى ترغب فى التعامل معها فى أثناء استخدام اللغة.

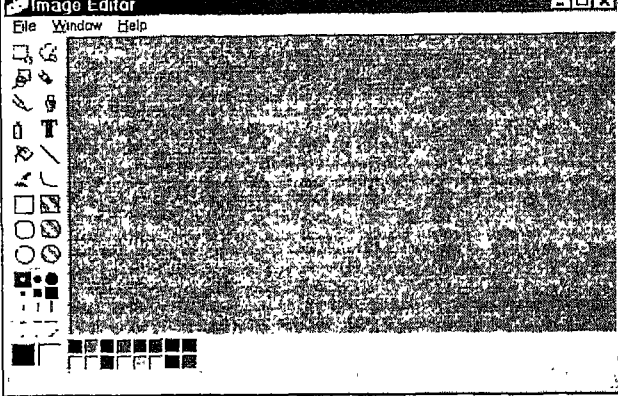
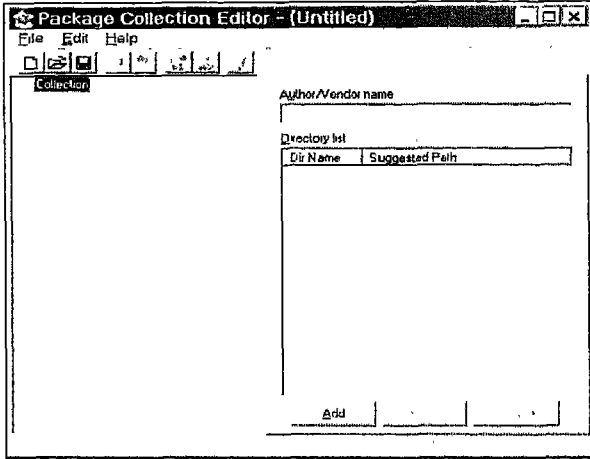


الجدول التالى يقدم لنا وصف مختصر للأوامر الافتراضية Default الموجودة بهذا

الجزء من القائمة Tools :

الوصف والاستخدام	الأمر
<p>عند اختيار هذا الأمر يتم فتح النافذة Database Desktop الموضح فى الشكل التالى :</p>  <p>وهذه النافذة تمثل أداة يمكن من خلالها إنشاء أو إعادة بناء جداول قواعد البيانات كما يمكن أيضا استعراض البيانات المخزنة فى هذه الجداول والتعديل بها أيضا. وأنت تستطيع التعامل مع الجداول الموجودة بقواعد البيانات المصممة من خلال Paradox و dBASE و SQL. هذا ونود هنا القول بأن هذه الأداة توجد فى الأصل فى المسار التالى :</p> <p>Program Files\Common Files\Borland Shared\Database Desktop</p>	Database Desktop
<p>من خلال هذا الأمر يمكن التعامل مع أداة لإنشاء وتحرير الصور التى يتم استخدامها فى التطبيق الذى تعده. وهذه الأداة تتمثل فى النافذة Image Editor الموضحة فى الشكل التالى :</p>	Image Editor



	
<p>يعمل هذا الأمر على فتح النافذة Package Collection Editor الموضحة في الشكل التالي :</p>  <p>هذه النافذة تسمح لك بأن تشاهد وتحرر الحزم البرمجية والملفات الأخرى الملحقة بأى من مجموعات الحزم البرمجية وبهذه الطريقة يمكن توزيع الحزم البرمجية لكي يستخدمها المبرمجون الآخرون.</p>	<p>Package Collection Editor</p>
<p>عند اختيار هذا الأمر يتم فتح النافذة XML Mapper Tool الموضحة في الشكل التالي :</p>	<p>XML Mapper</p>



من خلال هذه النافذة تستطيع أن تقوم بتعريف mappings التي تتم بين المستندات المعدة بلغة XML وحزم البيانات التي تستخدمها مجموعات البيانات العميلة client datasets.

الفصل الثالث

استخدام

مكتبة المكونات Components

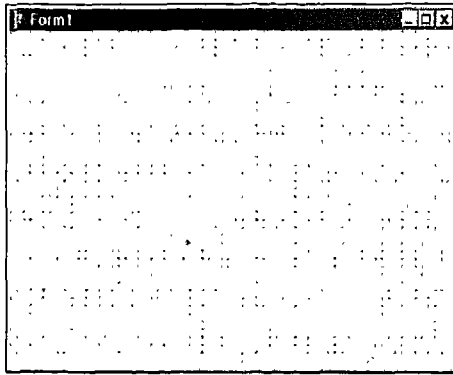
لدى Delphi 6



بيئة التطوير المتكاملة IDE

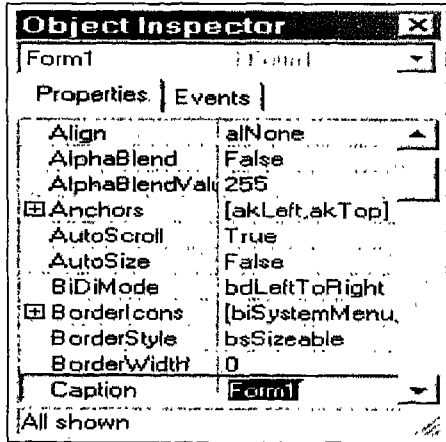
عندما تبدأ في تشغيل لغة Delphi فإنك تتواجد بشكل مباشر داخل بيئة التطوير المتكاملة IDE الخاصة بلغة Delphi. وبيئة العمل هذه تعمل على توفير كافة الأدوات التي تحتاج لها لتصميم وتطوير واختبار وفحص Debug ونشر وتوزيع التطبيقات.

تتضمن بيئة التطوير المتكاملة IDE الخاصة بلغة Delphi على أداة ظاهرية لتصميم الفورم والموضحة بالشكل التالي :



شكل توضيحي :

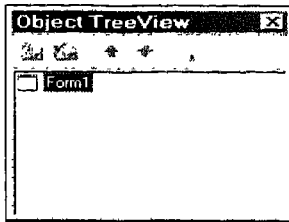
ونافذة لفحص الكائنات تسمى Object Inspector والموضحة بالشكل التالي :



شكل توضيحي :

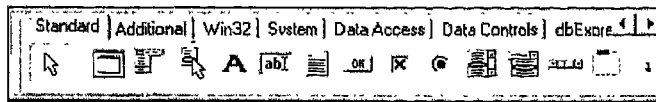
ونافذة للمشاهدة الشجرية للكائنات تسمى Object TreeView والموضحة بالشكل

التالي :



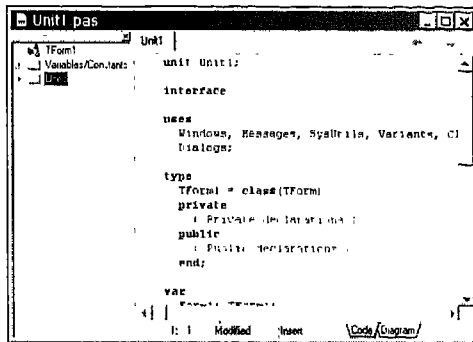
شكل توضيحي :

وبالليقة المكونات والموضحة بالشكل التالي :



شكل توضيحي :

ومحرر الكود البرمجي Code Editor الموضح بالشكل التالي :



شكل توضيحي :

ومدير المشروع Project Manager وأداة لإكتشاف الأخطاء البرمجية ومعالجتها
تسمى debugger بالإضافة إلى العديد من الأدوات الأخرى.

بعض الأدوات قد لا تكون موجودة بكافة إصدارات Delphi 6.



وأنت تستطيع التنقل بحرية من التمثيل المرئي لأي كائن (موجود في أداة تصميم
الفرم) إلى النافذة Object Inspector وذلك للتعديل في الحالة التشغيلية الابتدائية لهذا
الكائن ثم تنتقل بعد ذلك إلى محرر الكود البرمجي Code Editor للتعديل في منطق التنفيذ
لهذا الكائن.



التغيير في الخصائص المرتبطة بالكود البرمجي—مثل اسم أداة معاملة الحدث—في النافذة Object Inspector يؤدي على الفور إلى تغيير الكود البرمجي المناظر لهذه الخصائص. بالإضافة لذلك فإن التغييرات التي تجرى على الكود البرمجي—مثل تغيير أسلوب أداة معاملة الحدث في جملة الاعلان عن قطاع فورمة—تنعكس مباشرة داخل النافذة Object Inspector.

تعمل بيئة التطوير المتكاملة IDE على تدعيم إمكانية تطوير التطبيقات من خلال مراحل الدورة العمرية للمنتج—بداية من التصميم إلى التطوير. كما أن استخدام الأدوات المتوفرة لدى بيئة التطوير المتكاملة IDE يسمح لك بأن تعد نموذج حقيقي وفعل سريعا مما يؤدي بطبيعة الحال إلى التقليل من وقت الإعداد والتطوير.

تصميم التطبيقات

تمتلك لغة Delphi كافة الأدوات الضرورية للبدء في تصميم التطبيقات ومن بين هذه الأدوات ما يلي :

- نافذة فارغة تعرف بالفورمة form والتي يتم فيها تصميم واجهة الاستخدام User Interface (UI) للتطبيق الذي تعده.
- مجموعة هائلة من مكتبات القطاعات التي تشتمل على كمية كبيرة جدا من الكائنات التي يمكن استخدامها أكثر من مرة.
- النافذة Object Inspector والتي يمكن من خلالها التحكم في صفات وسلوك أي كائنات من خلال مجموعة من الخصائص والأحداث المرتبطة بهذا الكائن.
- محرر الكود البرمجي Code Editor الذي يمكن من خلاله التحكم في منطق عمل التطبيق في مرحلة التشغيل Run-Time.
- مدير المشروع Project Manager لإدارة الملفات التي يتألف منها مشروع واحد أو عدة مشروعات.
- العديد من الأدوات الأخرى مثل محرر الصور الموجود بشرط الأدوات وأداة معالجة الأخطاء المتكاملة ID (Integrated Debugger) الموجودة بالقوائم لتدعيم عملية تطوير التطبيق في بيئة التطوير المتكاملة IDE.



أدوات الأوامر الخطية Command-Line والتي تتضمن المترجمات compilers وأدوات الربط Linkers بالإضافة إلى العديد من الخدمات الأخرى.

تستطيع أن تستخدم لغة Delphi لتصميم أى نوع من التطبيقات التى تعمل بنظم التشغيل الـ ٣٢ بت-بداية من الخدمات العامة إلى البرامج المعقدة للوصول للبيانات والتعامل مع قواعد البيانات بالإضافة إلى التطبيقات الموزعة والتي تعمل بشبكات الحاسب.

أدوات قواعد البيانات وكذلك مكونات الارتباط بالبيانات التى تمتلكها لغة Delphi تسمح لك بأن تقوم سريعا بإعداد تطبيقات قواعد بيانات سواء كانت مكتبة Desktop (تعمل بأجهزة كمبيوتر مستقلة عن بعضها البعض) أو كانت شبكية Client/Server (التي تعمل بأجهزة كمبيوتر مرتبطة ببعضها البعض من خلال شبكة من الطراز Client/Server).

عن طريق استخدام أدوات التحكم الارتباط بالبيانات التى تمتلكها لغة Delphi تستطيع أن تشاهد على الطبيعة البيانات فى أثناء قيامك بتصميم التطبيق الخاص بك كما تشاهد على الفور نتائج عمليات الاستفسار التى تجرى بقواعد البيانات وكذلك التغييرات التى تجرى على واجهة استخدام التطبيق.

العديد من الكائنات المتوفرة فى مكتبة القطاعات يمكن الوصول إليها من خلال بيئة التطوير المتكاملة IDE وذلك من بالليته المكونات التى تقدم كافة أدوات التحكم سواء التى تكون ظاهرة فى مرحلة التشغيل Run-Time أو التى تكون مخفية فى مرحلة التشغيل Run-Time وكلاهما يمكن وضعه فى فورمة. ونود هنا القول بأن كل تبويب فى بالليته المكونات يشتمل على مجموعة من المكونات المتشابهة فى الأداء الوظيفي. والعموم نجد أن أسماء الكائنات الموجودة فى مكتبات القطاع تبدأ بالحرف T مثل اسم الكائن TStatusBar.

إحدى الأشياء التى تعتبر من المظاهر القوية والفعالة التى تتمتع بها لغة Delphi تتمثل فى إنه يمكنك إنشاء المكونات الخاصة بك وذلك باستخدام object Pascal. وفى هذا الصدد نقول إن أغلب المكونات التى تأتى مع لغة Delphi قد تم كتابتها من خلال Object Pascal. وأنت تستطيع إضافة المكونات التى تكتبها بنفسك إلى بالليته المكونات كما تستطيع أيضا تفصيل بالليته تضع فيها المكونات التى تكتبها وذلك عن طريق ضم تبويبات جديدة إلى بالليته المكونات.



تستطيع أيضا استخدام لغة Delphi لإعداد تطبيقات تكون لديها القدرة على العمل بمختلف نظم التشغيل مثل الويندوز وLinux وذلك عن طريق استخدام CLX التي تشمل على مجموعة من القطاعات التي لو تم استخدامها بدلا من القطاعات الموجودة في VCL فإنها تسمح للبرنامج الذي تعده بأن يعمل بكل من بيئة الويندوز ونظام التشغيل Linux.

استخدام مكتبة المكونات

في البداية نقول أن كل من VCL وCLX عبارة عن مكتبات قطاعات تتألف من مجموعة من الكائنات وبعض منها يتألف من مجموعة من المكونات أو أدوات التحكم والتي تستخدمها عند تطوير التطبيقات. وكلا المكتبتين تتشابهان معا إلى حد كبير وكلاهما يشتمل على العديد من الكائنات المتماثلة. وهناك بعض الكائنات الموجودة في VCL تعمل على تنفيذ مظاهر وإمكانات تكون متاحة ببيئة الويندوز فقط مثل الكائنات التي تظهر في كل من التبويبات التالية وبالليقة المكونات :

التبويب	أيقونات المكونات الموجودة به
ADO	Data Access Data Controls dbExpress DataSnap BDE ADO InterBa...
BDE	Data Access Data Controls dbExpress DataSnap BDE ADO InterBa...
QReport	InternetExpress Internet WebSnap FastNet Decision Cube QReport Di...
COM+	Decision Cube QReport Dialogs Win 3.1 Samples ActiveX COM+ Ind...
Web Services	WebServices InternetExpress Internet WebSnap FastNet Decision Cube...
Servers	Samples ActiveX COM+ Indu Clients Indu Servers Indu Misc Servers ...



أما بالنسبة للكائنات الموجودة في CLX فهي متاحة بكل من بيئة الويندوز ونظام التشغيل Linux.

الكائنات الموجودة بكل من VCL و CLX تعتبر كيانات نشطة Active entities وتشتمل على كافة البيانات الضرورية والأساليب (الكود) الذى يعمل على تعديل هذه البيانات التى تكون مخزنة فى حقول وخصائص الكائنات كما أن الكود يكون مؤلف من الأساليب التى تتعامل مع قيم الحقول والخصائص. هذا ويتم الاعلان عن كل كائن على أساس أنه قطاع Class. ونود هنا القول بأن كافة الكائنات الموجودة بكل من VCL و CLX تنحدر من الكائن TObject الذى يعتبر الجد الأسمى لهذه الكائنات بما فيها الكائنات التى تقوم بإعدادها بنفسك من خلال Object Pascal.

أى مجموعة فرعية من الكائنات يطلق عليها مكونات. ويمكن تعريف المكونات بأنها كائنات تستطيع وضعها فى فورمة أو Module بيانات والتعامل معها فى أثناء مرحلة التصميم Design-Time. هذا وتظهر المكونات فى باليئة المكونات. وأنت تستطيع توصيف خصائص هذه المكونات بدون استخدام أى كود برمجى. ونود هنا القول بأن كافة المكونات بكل من VCL و CLX تنحدر من الكائن TComponent.

المكونات تعتبر كائنات لو نظرنا إليها من منظور أسلوب البرمجة OOP (اختصار للمصطلح Object-Oriented Programming) وذلك لأنها :

❶ تتمتع بمجموعة من الدوال التى يتم استخدامها للتعامل مع البيانات والوصول إليها بقواعد البيانات.

❷ تتوارث كل من البيانات وصفات السلوك من الكائنات التى تنحدر منها.

❸ تعمل بطريقة تفاعلية وتبادلية مع الكائنات التى تنحدر من نفس الجد وذلك فى إطار منظومة يطلق عليها polymorphism (تعدد الأشكال).

باختلاف معظم المكونات نجد أن الكائنات لا تظهر فى باليئة المكونات. ولكن بدلا من ذلك يتم الاعلان عن متغير حالة افتراضية وهذا الاعلان يكون فى وحدة الكائن ويتم تلقائيا ولكن إذا لم يتم تلقائيا فإنه فى هذه الحالة ينبغى عليك القيام به بنفسك.



أدوات التحكم تعتبر نوعا خاصا من المكونات وتكون ظاهرة للمستخدمين فى مرحلة التشغيل Run-Time. هذا وتعتبر أدوات التحكم مجموعة فرعية من المكونات. ويمكن القول بأن أدوات التحكم عبارة عن مكونات مرئية تستطيع أن تشاهدها عندما يكون التطبيق الذى تتولى إعداده فى حالة عمل. وكافة أدوات التحكم تمتلك عدد من الخصائص التى يمكن من خلالها تحديد فى الصفات الظاهرية لأدوات التحكم مثل الارتفاع Height والعرض Width. ونود هنا القول بأن كل من الخصائص والأساليب والأحداث التى تمتلكها كافة أدوات التحكم يتم توارثها من TControl.

الخصائص والأساليب والأحداث

كل من VCL و CLX تعمل على تكوين هياكل شجرية للكائنات المرتبطة ببيئة التطوير المتكاملة IDE الخاصة بلغة Delphi حيث تستطيع إعداد وتطوير التطبيقات بشكل سريع وفعال. والكائنات الموجودة بكلا المكتبتين تكون معتمدة على مجموعة من الخصائص والأساليب والأحداث. فكل كائن يتضمن عدد من الخصائص Properties التى تعرف بأنها أعضاء بيانات كما يمتلك عدد من الدوال التى تعمل على هذه البيانات وهذه الدوال يطلق عليه أساليب methods كما أن كل كائن لديه عدد من الطرق التى تتحكم فى تفاعله مع المستخدم وهذه الطريق تعرف بأنها أحداث Events.

نود هنا القول بأن VCL قد تم كتابتها Object Pascal فى حين أن CLX يعتمد على QT وهو عبارة عن مكتبة قطاعات معدة بلغة C++.



الخصائص Properties

الخصائص Properties عبارة عن صفات أى كائن وهذه الصفات تؤثر بشكل مباشر سواء فى السلوك الظاهرى للكائن أو فى العمليات التى يجريها الكائن. فعلى سبيل المثال الخاصية Visible تعمل على تحديد ما إذا كان الكائن ظاهرا أم سيكون مختفيا بواجهة استخدام التطبيق فى أثناء تشغيل التطبيق. وفى هذا الصدد نقول إن الخصائص المصممة جيدا تجعل من السهولة بمكان للآخرين استخدام المكونات التى تضعها بالتطبيق كما تجعل أيضا من السهل عليك أن تجرى أى تعديلات ترغبها على هذه المكونات.



- فيما يلي سنستعرض سويا المظاهر والإمكانات المفيدة للخصائص :
- ❖ تختلف الخصائص عن الأساليب التي تكون متاحة فقط في مرحلة التشغيل Run-Time فأنت تستطيع مشاهدة وتغيير قيم الخصائص في مرحلة التصميم والحصول مباشرة على تغذية مرتدة كلما تغيرت الكائنات في بيئة التطوير المتكاملة IDE.
 - ❖ يمكن الوصول للخصائص من خلال النافذة Object Inspector حيث يمكنك إجراء أى تعديلات ترغبها على قيم خصائص الكائن الذى تتعامل معه. ونود هنا القول بأن تحديد قيم الخصائص في مرحلة التصميم تكون أسهل من تحديدها عن طريق كود برمجى كما يجعل من الأسهل التعديل فى الأكواد البرمجية الموجودة بالتطبيق حيث ستكون الأكواد البرمجية أصغر حجما فى هذه الحالة.
 - ❖ بسبب أن البيانات موجودة داخل الخصائص فإن ذلك يوفر لها نوع من الحماية والخصوصية بحيث تكون مرتبطة بالكائن ذاته.
 - ❖ الاستدعاءات الحقيقية التى تتم سواء للحصول على قيم الخصائص أو لتعديلها يطلق عليها أساليب ومن ثم يكون فى الإمكان إجراء عملية معالجة خاصة للكائن بعيدا عن أعين المستخدمين للتطبيق فى مرحلة التشغيل Run-Time. فعلى سبيل المثال يمكن وضع بيانات فى جدول ولكن قد تبدو للمبرمج كما لو كانت عضو بيانات عادى.
 - ❖ تستطيع أن تعد منهم منطقى يعمل على استثارة الأحداث أو تعديل بيانات أخرى فى أثناء الوصول للخاصية. فعلى سبيل المثال عملية تغيير قيمة خاصية واحدة قد تتطلب تعديل قيمة خاصية أخرى وهكذا...وأنت تستطيع إجراء التغيير فى قيمة الخاصية عن طريق الأساليب المعدة خصيصا لهذا الغرض.
 - ❖ يمكن أن تكون الخصائص تصويرية Virtual.
 - ❖ هناك بعض الخصائص التى لا تكون مخصصة لكائن واحد فقط. فتغيير قيمة خاصية واحدة لأحد الكائنات يمكن أن يؤثر فى العديد من الكائنات الأخرى. فعلى سبيل المثال تحديد قيمة الخاصية Checked للكائن radio button يؤثر على الكائنات الأخرى التى من نفس النوع والموجودة معا فى مجموعة واحدة.

الأساليب Methods

الأسلوب method عبارة عن إجراء يكون دوما ملحقا بقطاع. والأساليب هي المسئولة عن تحديد سلوك أى كائن. وفي هذا الصدد نقول إن الأساليب الخاصة بأى قطاع يمكنها الوصول لكافة الخصائص سواء كانت عامة public أو محمية protected أو خاصة Private كما يمكنها الوصول أيضا لأعضاء البيانات للقطاع كما إنه من المعتاد أن يشار إليها على أساس كونها دوال member functions.

الأحداث Events

الحدث Event عبارة عن فعل أو واقعة يتحدد وقوعها من خلال البرنامج. وأغلب التطبيقات الحديثة يقال عنها أنها من الطائفة التى تعرف بـ Event-driven وذلك لأنه تم تصميم هذه التطبيقات بحيث تستجيب للأحداث التى تقع أثناء تفاعل المستخدمين معها. ففى أى برنامج لا يكون للمبرمج أى سبيل للتنبؤ بالتتابع المضبوط لوقوع الأحداث فى أثناء تعامل المستخدم مع البرنامج. فقد يقوم المستخدم باختيار عنصر من أحد القوائم الموجودة بالبرنامج أو قد يقوم بالنقر بالماوس على أحد المفاتيح أو الأيقونات أو قد يقوم بالتعليم على نص ما وهكذا... ولكن على العموم تستطيع كتابة كود برمجى لمعاملة الأحداث التى تربطها بالعناصر الموجودة بالبرنامج والتى تقع عند تعامل المستخدم مع هذه العناصر وذلك بدلا من أن تكتب كود برمجى دوما يتم تنفيذه بتتابع أو ترتيب ثابت.

بغض النظر عن الكيفية التى يتم بها استدعاء أى حدث فإن نركز على أن لغة Delphi تلقى نظرة لكى تشاهد لو أنك قمت بكتابة أى كود برمجى لمعاملة حدث ما أم لا. ولو كان ذلك حدث فإن يتم تنفيذ هذا الكود البرمجى ولكن خلاف ذلك تجد أنه يتم التعامل مع الحدث بالطريقة الطبيعية أو الافتراضية.

أنواع الأحداث

أنواع الأحداث التى يمكن أن تحدث يمكن تصنيفها إلى صنفين أساسيين هما :

● أحداث المستخدم User Events.

● أحداث النظام System Events.

وكما قلنا سابقا فإنه بغض النظر عن طريقة استدعاء الحدث تقوم لغة Delphi برؤية ما إذا كنت قد خصصت أى كود برمجى لمعاملة هذا الحدث أم لا. ولو أنك قمت بذلك فى هذه الحالة يتم تنفيذ هذا الكود البرمجى عند وقوع الحدث وإلا فلن يحدث شيء عند وقوع الحدث.

أحداث المستخدم User Events

أحداث المستخدم User Events عبارة عن الأفعال التى تتم من خلال مستخدم البرنامج. والأمثلة على أحداث المستخدم User Events كثيرة ومتعددة منها الحدث OnClick الذى يقع عندما يقوم المستخدم بالنقر بالماوس على أى عنصر بالبرنامج وكذلك الحدث OnKeyPress والذى يقع عند الضغط المستخدم على أى مفتاح بلوحة المفاتيح بالإضافة إلى الحدث OnDblClick والذى يقع عندما ينقر المستخدم بالماوس نقرا مزدوجا. وهذه الأحداث غالبا ما تكون مرتبطة بالأفعال التى يفعلها مستخدم البرنامج.

أحداث النظام System Events

أحداث النظام System Events عبارة عن الأحداث التى تقع بواسطة نظام التشغيل نفسه. فعلى سبيل المثال الحدث OnTimer الذى يقع بواسطة المكون Timer حينما تنقضى فترة زمنية محددة مسبقا أما الحدث OnCreate فيقع عندما يتم إنشاء مكون فى حين أن الحدث OnPaint يقع عندما يكون هناك مكون أو نافذة فى حاجة لأن يتم إعاده رسمها وهكذا...ومن المعتاد أن لا تقع أحداث النظام System Events من خلال الأفعال التى يفعلها مستخدم البرنامج.

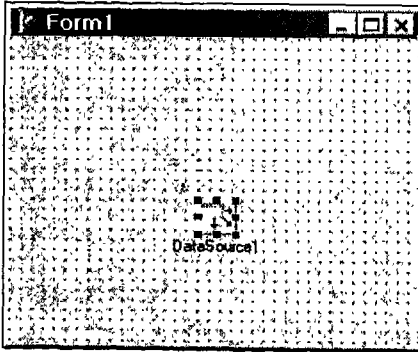
الـ Object Pascal ومكتبات القطاعات

الـ Object Pascal عبارة عن مجموعة من الامتدادات للغة Pascal القياسية التى تعتبر اللغة التى بنيت عليها لغة Delphi. وعن طريق استخدام كل من باليئة المكونات والنافذة Object Inspector الخاصة بلغة Delphi تستطيع وضع مكونات VCL أو مكونات CLX بالفورمة وتتعامل مع خصائص كل منهم بدون كتابة أى كود برمجى.

كافة الكائنات تنحدر من TObject وهو عبارة عن قطاع استخلاص يمتلك عدد من الأساليب تتحكم فى السلوك الوظيفى مثل البناء والهدم والتعامل مع الرسائل. كما أن TObject يصبح على الفور جد للعديد من القطاعات البسيطة.

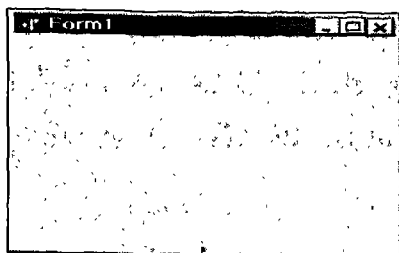
المكونات الموجودة فى VCL أو CLX تنحدر من قطاع الاستخلاص TComponent. ونود هنا القول بأن المكونات عبارة عن كائنات تستطيع التعامل معها بالفورم فى مرحلة التصميم. هذا والمكونات المرئية-مثل المكونات TForm و TSpeedButton والتى تظهر على الشاشة فى مرحلة التشغيل Run-Time-يطلق عليها أدوات تحكم وهى تنحدر من TControl.

بالإضافة إلى مجموعة المكونات المرئية هناك مكتبات مكونات تشتمل على العديد من الكائنات الغير مرئية. ونود هنا القول بأن بيئة التطوير المتكاملة IDE تسمح لك بأن تضيف العديد من المكونات الغير مرئية إلى البرامج التى تعدها بنفسك وذلك عن طريق اسقاطهم فى الفورم. فعلى سبيل المثال لو أنك تكتب تطبيق يتصل بقاعدة بيانات فى هذه الحالة قد تحتاج لأن تضع المكون TDataSource بفورمة. هذا وبالرغم من أن المكون TDataSource غير مرئى إلا إنه يتم تمثيله بالفورمة بأيقونة كما هو موضح بالشكل التالى :



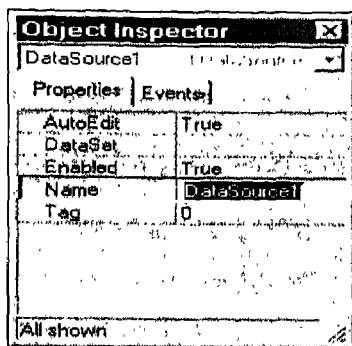
شكل توضيحي :

فى حين أن هذا المكون لا يكون ظاهرا بالفورمة فى مرحلة التشغيل Run-Time كما هو موضح بالشكل التالى :

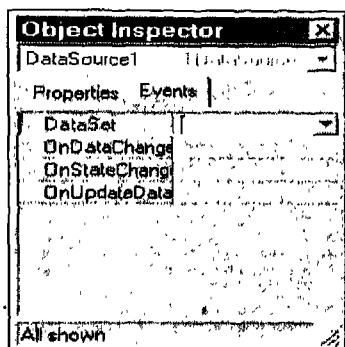


شكل توضيحي :

وأنت تستطيع التعامل مع الخصائص والأحداث الخاصة بالمكون TDataSource في النافذة Object Inspector (كما هو موضح بالشكلين التاليين) كما هو الحال مع المكونات المرئية :



شكل توضيحي :



شكل توضيحي :

عندما تكتب قطاعات خاصة بك من خلال Object Pascal فينبغي أن تكون منحدرة من TObject في مكتبة القطاع التي تخطط أن تستخدمها. هذا عليك أن تستخدم VCL لو أنك تكتب تطبيق سيعمل بيئة الويندوز فقط في حين عليك أن تستخدم CLX لو كنت تكتب تطبيقات تعمل بكل من بيئة الويندوز ونظام التشغيل Linux. وعن طريق

إعداد قطاعات جديدة من قطاع التأسيس المناسب (أو واحد من القطاعات التي نتحدر منه) فإنك توفر القطاعات الخاصة بك لها أداء وظيفي محدد ولكن ينبغي عليك التأكد من أن هذه القطاعات تعمل بشكل جيد مع القطاعات الأخرى الموجودة في مكتبة القطاع.

استخدام نموذج الكائن Object Module

أسلوب البرمجة OOP يعتبر الامتداد الطبيعي لأسلوب البرمجة الهيكلية الذي يركز في المقام الأول على إعداد كود برمجي يمكن استخدامه أكثر من مرة كما يهدف أيضا إلى التكامل بين البيانات والأداء الوظيفي. وبمجرد أن تقوم بإنشاء كائن (أو قطاع حيث ذلك أكثر شيوعا الآن) فإنك تستطيع أنت والمبرمجين الآخرين استخدام هذا الكائن أو القطاع في مختلف التطبيقات مما يؤدي إلى التقليل بقدر الإمكان من فترة إعداد التطبيقات وفي نفس الوقت يعمل على زيادة مستوى الإنتاجية لدى المبرمج.

ما هو الكائن Object

الكائن Object أو القطاع Class عبارة عن وعاء أو مخزن للبيانات والعمليات التي تجري عليها في وحدة واحدة. وفي هذا الصدد نقول أنه قبل إبتكار أسلوب البرمجة OOP كانت كل من البيانات والعمليات التي تجري عليها (الدوال Functions) يتم التعامل معها على أساس أنها عناصر منفصلة عن بعضها البعض.

تستطيع أن تبدأ في فهم الكائنات لو أنك فهمت سجلات كائن الباسكال أو الهياكل البنائية بلغة C. فالسجلات تتألف من حقول تشتمل على ألبينات حيث أن كل حقول ينتمي لنوع معين. والسجلات تجعل من السهولة بمكان الإشارة إلى مجموعة من عناصر بيانات مختلفة في النوع.

كذلك يمكن اعتبار الكائنات بمثابة مجموعات من عناصر البيانات. ولكن الكائنات-على خلاف السجلات-تكون مشتملة على عدد من الإجراءات والدوال التي تتعامل مع عناصر البيانات هذه. وهذه الإجراءات والدوال يطلق عليها أساليب Methods.

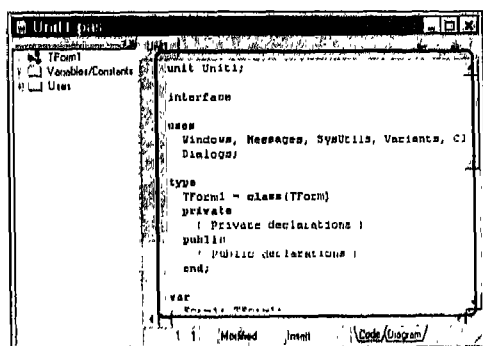


عناصر البيانات الموجودة بالكائن يمكن الوصول إليها عن طريق الخصائص الخاصة بالكائن. وخصائص الكائنات VCL و CLX لديها قيم يمكنك تغييرها في مرحلة التصميم بدون كتابة كود برمجى للقيام بهذا التغيير. ولو أنك تحتاج لأن يتم تغيير قيمة خاصة في مرحلة التشغيل Run-Time في هذه الحالة ستحتاج لأن تكتب كود برمجى صغير الحجم للغاية.

الدمج بين البيانات والأداء الوظيفى فى وحدة واحدة يطلق عليه التغليف encapsulation أو التكامـل . هذا وبالإضافة إلى مبدأ التغليف أو التكامـل نجد أن أسلوب البرمجة OOP يتصف بصفة التوارث inheritance وبصفة التعددية polymorphism. ونود هنا القول بأن التوارث يعنى أن الكائنات تكتسب الأداء الوظيفى الخاص بها من كائنات أخرى (يطلق عليها الأسلاف ancestors) كذلك فإن الكائنات تستطيع تعديل سلوكها التى ورثته من الكائنات الأخرى. أما صفة التعددية فتعنى أن الكائنات المختلفة والتى تنحدر من نفس السلف تعمل على تدعيم نفس التفاعل مع الخصائص والأساليب والذى يطلق عليه فى الغالب التفاعلية التبادلية interchangeably.

اكتشاف الكائنات المتاحة لدى لغة Delphi

عندما تنشأ مشروع جديد تجد أن لغة Delphi تقدم لك فورمة جديدة لكى تقوم بتفصيلها. كما إنه فى محرر الكود البرمجى Code Editor تقوم لغة Delphi بالإعلان عن نوع قطاع جديد للفورمة وتعمل أيضا على إعداد الكود البرمجى الذى يتولى مهمة إنشاء حالة جديدة للفورمة كما هو موضح بالشكل التالى :



شكل توضيحي :

الكود البرمجي الذي يتم تكوينه للتطبيق الجديد الذي يتم إعداده للعمل ببيئة الويندوز فقط يكون مشابه إلى حد كبير للكود البرمجي التالي :

```
unit Unit1;
interface
uses Windows, Classes, Graphics, Forms, Controls, Dialogs;
type
  TForm1 = class(TForm)    { The type declaration of the form
                           begins here }
  private
    { Private declarations }
  public
    { Public declarations }
  end ;                    { The type declaration of the form ends here }
var
  Form1: TForm1;
implementation            { Beginning of implementation part }
{$R *.DFM}
end.                      { End of implementation part and unit }
```

نوع القطاع الجديد عبارة عن TForm1 وهو منحدر من النوع TForm الذي يعتبر أيضا قطاع.



القطاع يشبه السجل على أساس أن كل منهما يشتمل على حقول بيانات ولكن القطاع يشتمل أيضا على أساليب—عبارة عن كود برمجي يؤثر في البيانات المخزنة بالكائن. وحتى الآن يظهر TForm1 بحيث لا يشتمل على أى حقول أو أساليب وذلك لأنك لم تقم بعد بإضافة أى مكونات للفورمة (حقول الكائن الجديد) كما إنك لم تقم بإنشاء أى أدوات معاملة للأحداث (الأساليب الخاصة بالكائن الجديد). ولكن بالرغم من ذلك فإننا نقول إن TForm1 يشتمل على حقول وأساليب متوارثة حتى ولو لم تكن ترى أى منهم في الكود البرمجي السابق.


فيما يلي جملة إعلان عن متغير يسمى Form1 ينتمي للنوع الجديد TForm1 .:

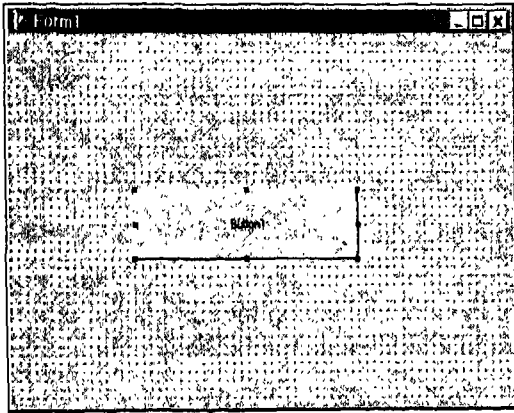
```
Var
  Form1: TForm1;
```

المتغير Form1 يمثل حالة instance أو كائن ينتمي لنوع القطاع TForm1. وأنت تستطيع الاعلان عن أكثر من حالة تنتمي لنوع قطاع وقد تجبر على القيام بذلك في بعض الأحيان. فعلى سبيل المثال تحتاج لأن تعلن عن أكثر من حالة لكي تنشأ عدة نوافذ وليدة في التطبيقات التي تشتمل على واجهة استخدام من النوع MDI (اختصار للمصطلح Multiple Document Interface). وفي هذا الصدد نقول إن كل حالة تحتفظ بالبيانات الخاصة بها ولكن في نفس الوقت كافة الحالات تستخدم نفس الكود البرمجي لتنفيذ الأساليب الخاصة بها.

بالرغم من أنك لم تقم بعد بإضافة أى مكونات إلى الفورمة كما لم تقم بكتابة أى كود برمجي حتى الآن إلا أنك تمتلك الآن تطبيق يعتبر كامل وتستطيع ترجمته Compile وتشغيله ولكن كل الذى سيظهر على الشاشة عند التشغيل عبارة عن فورمة فارغة فقط.

لتفترض أنك أضفت المكون Button إلى هذه الفورمة (بالنقر المزدوج بالماوس على

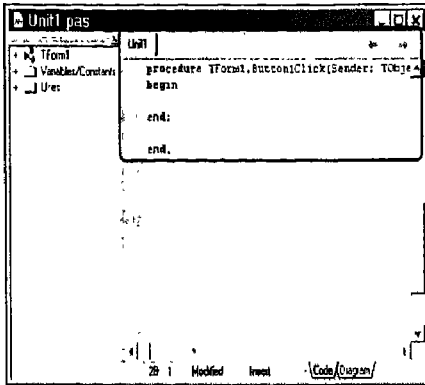
أيقونة المكون Button  بباليئة المكونات) لتصبح كما هو موضح بالشكل التالى :



شكل توضيحي :

وترغب الآن فى كتابة الكود البرمجي لأداة معاملة الحدث OnClick الخاص بالمكون Button والذى يعمل على تغيير لون الفورمة لتصبح خضراء عندما يقوم المستخدم بالنقر بالماوس على المكون Button فى مرحلة التشغيل Run-Time. وللقيام بذلك اتبع الخطوات التالية :

- (١) انقر بالماوس نقرًا مزدوجًا على المكون Button بالفورمة ليظهر على الشاشة محرر الكود البرمجي Code Editor وبه أداة المعاملة للحدث OnClick الخاص بهذا المكون كما هو موضح بالشكل التالي :

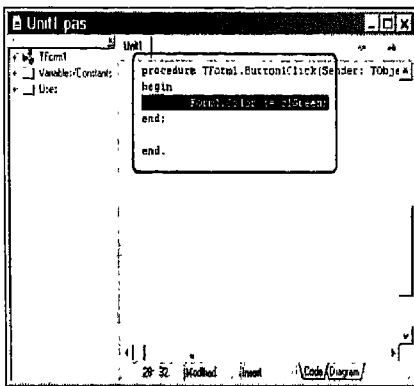


شكل توضيحي :


- (٢) بالسطر الموجود به مؤشر الكتابة اكتب الكود التالي :

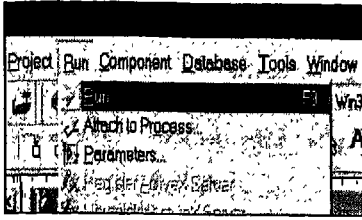
Form1.Color := clGreen;

كما هو موضح بالشكل التالي :



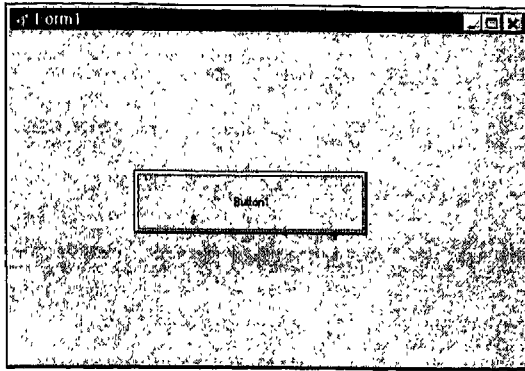
شكل توضيحي :

- (٣) والآن قم بتشغيل البرنامج بالضغط على المفتاح F9 بلوحة المفاتيح أو انقر بالماوس على الأيقونة  بشريط الأدوات أو افتح القائمة Run واختار منها Run كما هو موضح بالشكل التالي :



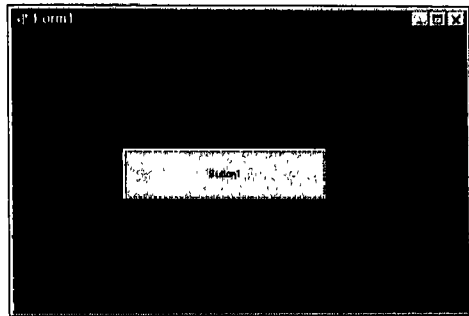
شكل توضيحي :

(٤) لتظهر على الشاشة الفورمة وهي مشتملة على المفتاح كما هو موضح بالشكل التالي :



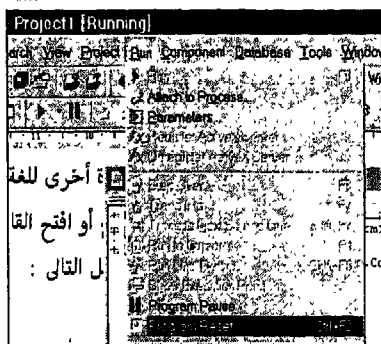
شكل توضيحي :

(٥) والآن انقر بالماوس على المفتاح Button1 لتجد أن الفورمة قد تغير لونها وأصبحت خضراء اللون كما هو موضح بالشكل التالي :



شكل توضيحي :

(٦) اغلق البرنامج بالنقر بالماوس على الزر (X) بالركن الأيمن العلوي للفورمة أو اضغط على المفاتيح Alt+F4 بلوحة المفاتيح أو عد مرة أخرى للغة Delphi وفيها اضغط على المفاتيح Ctrl+F2 بلوحة المفاتيح أو افتح القائمة Run واختر منها الأمر Program Reset كما هو موضح بالشكل التالي :



شكل توضيحي :

يمكن للكائنات أن تشتمل على كائنات أخرى على أساس أنها حقول بيانات. ففي كل مرة تقوم فيها بوضع مكون في فورمة يظهر على الفور حقل جديد في جملة الإعلان عن النوع بمحرر الكود البرمجي Code Editor. ولو أنك قمت بإنشاء التطبيق السالف الذكر ثم ألقيت نظرة على الكود البرمجي الموجود الآن في محرر الكود البرمجي Code Editor فسوف تشاهد الآتي :

```
unit Unit1;
interface
uses Windows, Classes, Graphics, Forms, Controls;
type
  TForm1 = class(TForm)
    Button1: TButton;      { New data field }
    procedure Button1Click(Sender: TObject);      { New
                                                    method declaration }
  private
    { Private declarations }
  public
    { Public declarations }
  end ;
var
  Form1: TForm1;
implementation
{$R *.DFM}
procedure TForm1.Button1Click(Sender: TObject);      { The code of
                                                    the new method }
begin
  Form1.Color := clGreen;
end ;
end.
```



نود هنا القول بأن الكائن TForm1 يمتلك حقل يسمى Button1 وهو يناظر المفتاح الذى أضفته إلى الفورمة قبل ذلك. وفى هذا الصدد نقول إن TButton عبارة عن نوع قطاع ومن ثم فإن Button1 يشير إلى كائن.

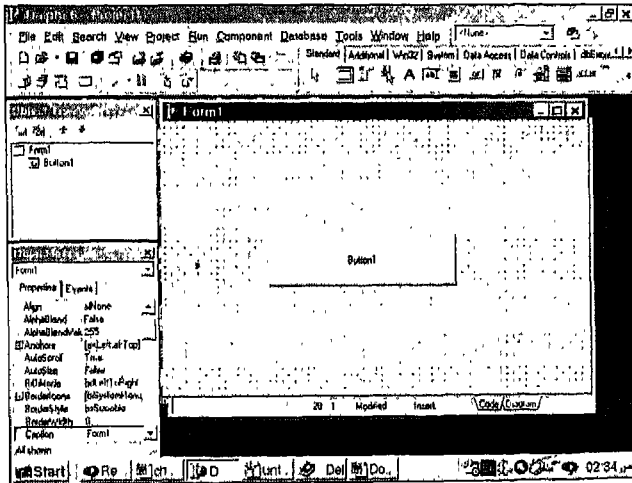
أدوات معاملة كافة الأحداث التى تكتبها فى لغة Delphi تعتبر بمثابة أساليب لكائن الفورمة. ففي كل مرة تقوم فيها بإنشاء أداة معاملة لحدث فإنه يتم على الفور الإعلان عن أسلوب فى نوع كائن الفورمة. والآن أصبح النوع TForm1 مشتملا على أسلوب جديد وهو الإجراء Button1Click والذى تم الإعلان عنه داخل جملة الإعلان عن النوع TForm1. أما الكود البرمجى الذى يقوم بتشغيل الأسلوب Button1Click فيظهر فى الجزء implementation من الوحدة.

تغيير اسم مكون

ينبغى عليك دوما استخدام النافذة Object Inspector لتغيير اسم أى مكون. فعلى سبيل المثال لنفترض أنك ترغب فى تغيير اسم فورمة من الاسم الافتراضى لها وهو Form1 ليصبح معبرا أكثر عن وظيفة الفورمة وليكن مثلا ColorBox. وللقيام بذلك اتبع الخطوات التالية :

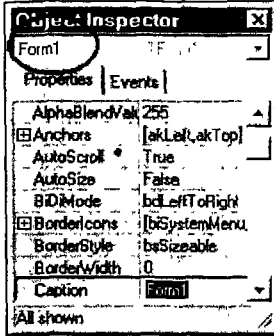
(١) اجعل الفورمة هى العنصر المختار فى بيئة التطوير المتكاملة IDE كما هو

موضح بالشكل التالى :

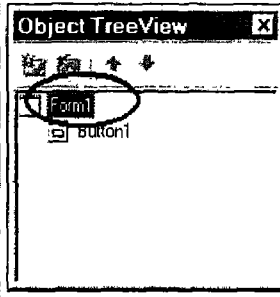


شكل توضيحي :

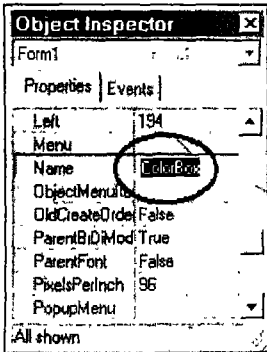
لكي تتأكد من أن الفورمة هي المختارة حالياً عليك أن تتأكد من أنك تشاهد Form1 بالقائمة المنسدلة الموجودة بقمة النافذة Object Inspector كما هو موضح بالشكل التالي :



كما تأكد أيضاً من إن Form1 معلم عليها في النافذة Object TreeView كما هو موضح بالشكل التالي :



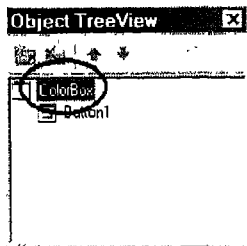
(٢) في النافذة Object Inspector وعند الخاصية Name اكتب ColorBox ثم اضغط على المفتاح Enter بلوحة المفاتيح كما هو موضح بالشكل التالي :



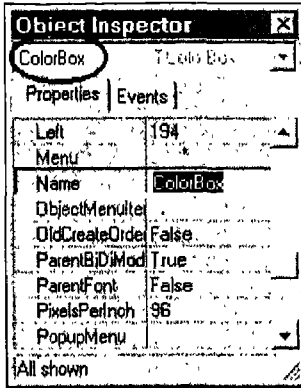
شكل توضيحي :



تلاحظ أن اسم الفورمة أصبح ColorBox في كل من النافذة Object TreeView كما هو موضح بالشكل التالي :



وبالقائمة المنسدلة الموجودة في أعلى النافذة Object Inspector كما هو موضح بالشكل التالي :



من مظاهر القوة التي تتمتع بها لغة Delphi أنك عندما تقوم بتغيير قيمة الخاصية Name للفورمة من خلال النافذة Object Inspector تجد أن الاسم الجديد ينعكس مباشرة داخل الملف dfm. أو الملف xfm. الخاص بالفورمة (في العادة لا يتم تعديل محتويات هذه الملفات يدوياً) وكذلك في الكود البرمجي الذي تكونه لغة Delphi وبالتالي سيصبح كالتالي :

```
unit Unit1;
interface
uses Windows, Classes, Graphics, Forms, Controls;
type
  TColorBox = class (TForm) { Changed from TForm1 to
                                TColorBox }
    Button1: TButton;
```

```

procedure Button1Click(Sender: TObject);
private
    { Private declarations }
public
    { Public declarations }
end ;

var
    ColorBox: TColorBox;      { Changed from Form1 to ColorBox }
implementation
    {$R *.DFM}
procedure TColorBox.Button1Click(Sender: TObject);
begin
    Form1.Color := clGreen;    { The reference to Form1 didn't
                                change! }
end ;
end.

```

تلاحظ أن الكود البرمجي الموجود في أداة معاملة الحدثOnClick المرتبط بالفتاح Button1 لم يحدث به أى تغيير. والسبب فى ذلك أنك قمت بنفسك بكتابة هذا الكود البرمجي وليست اللغة ولذلك ينبغى عليك تحديثه بنفسك وتصحح أى إشارات مرجعية للفورمة وبالتالي يصبح كالآتى :

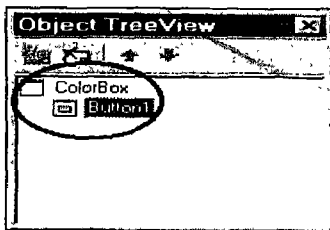
```

procedure TColorBox.Button1Click(Sender: TObject);
begin
    ColorBox.Color := clGreen;
end ;

```

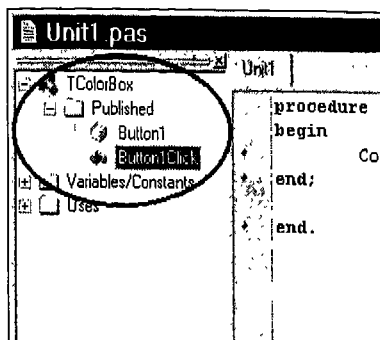
توارث البيانات والكود البرمجي من كائن

الكائن TForm1 الذى نتعامل معه حتى الآن يبدو أنه بسيطاً. فهذا الكائن يظهر لكى يشتمل على حقل واحد وهو الحقل Button1 كما نرى فى النافذة Object TreeView الموضحة فى الشكل التالى :



شكل توضيحي :

كما إنه يشتمل أيضا على أسلوب واحد وهو الأسلوب Button1Click كما نشاهد في الجزء الأيسر بنافذة محرر الكود البرمجي Code Editor وكما هو موضح بالشكل التالي :



شكل توضيحي :

فى حين أنه لا يشتمل على أى خصائص. ومن ثم فأنت تستطيع إظهار أو إخفاء الفورمة أو تغيير حجمها كما تستطيع أيضا إضافة أو مسح أيقونات الإطار القياسى بالإضافة إلى إمكانية تهيئة الفورمة بحيث تصبح جزء من تطبيق MDI. وأنت تستطيع القيام بهذه الأشياء لأن الفورمة قد ورثت كافة الخصائص والأساليب الخاصة بالمكون TForm.

عندما تضيف فورمة جديدة إلى المشروع الذى تعده فإنك تبدأ مع TForm وتعمل على تفصيله عن طريق إضافة مكونات له وتغيير قيم الخصائص الخاصة به وكتابة أكواد برمجية لأدوات معالجة الأحداث المرتبطة به.

على العموم لكى تقوم بتفصيل أى كائن عليك أولا أن تسوق كائن جديد من كائن موجود بالفعل فعندما تضيف فورمة جديدة إلى المشروع الذى تعده تقوم لغة Delphi تلقائيا بـ Drive فورمة جديدة من النوع TForm وذلك من خلال الكود التالى :

TForm1 = class (TForm)

الكائن المساق يرث كافة الخصائص والأحداث والأساليب الخاصة بالكائن الأصلى. ونود هنا القول بأن الكائن المساق يطلق عليه الكائن المنحدر descendant أو الخلف فى حين أن الكائن الأصلى يطلق عليه السلف ancestor.

أى كائن يمكن أن يكون له سلف (جد) واحد فقط وفى نفس الوقت يمكن أن يكون له العديد من الأحفاد (الخلف).





فيما يلي سنستعرض سويا كل من الخصائص والأحداث والأساليب الخاصة بالكائن الجد TForm وذلك من خلال الجدول التالي :

الأحداث	الأساليب	الخصائص
OnActivate	Cascade	Active
OnClick	Next	ActiveControl
OnClose	Previous	ActiveMDIChild
OnCloseQuery	Tile	BorderIcons
OnContextPopup	AfterConstruction	BorderStyle
OnCreate	BeforeDestruction	Canvas
OnDblClick	Close	ClientHandle
OnDeactivate	CloseQuery	ClientHeight
OnDestroy	Create	ClientRect
OnDragDrop	CreateNew	ClientWidth
OnDragOver	DefocusControl	DesignerHook
OnHelp	Destroy	DropTarget
OnHide	FocusControl	FocusedControl
OnKeyDown	Hide	FormState
OnKeyPress	InvokeHelp	FormStyle
OnKeyString	IsShortCut	Icon
OnKeyUp	Release	KeyPreview
OnLoaded	SetFocus	MDIChildCount
OnMouseDown	SetFocusedControl	MDIChildren
OnMouseMove	Show	Menu
OnMouseUp	ShowModal	ModalResult
OnMouseWheel	DisableAutoRange	Parent
OnMouseWheelDown	EnableAutoRange	PixelsPerInch
OnMouseWheelUp	ScrollInView	Position
OnPaint	ContentsRect	Scaled
OnResize	Broadcast	Visible
OnShortCut	CanFocus	WindowState
OnShow	ContainsControl	AutoScroll
	ControlAtPos	ChildHandle
	CreateParented	HorzScrollBar
	CreateParentedControl	VertScrollBar
	DisableAlign	Bitmap
	EnableAlign	Brush
	FindChildControl	ClientOrigin
	FlipChildren	ControlCount



	Focused GetTabOrderList HandleAllocated HandleNeeded InsertControl Invalidate InvalidateRect Realign RemoveControl Repaint ScaleBy ScrollBy Update UpdateControlState BeginDrag BringToFront ClientToParent ClientToScreen DragDrop Dragging EndDrag InitiateAction ParentToClient Refresh ScreenToClient SendToBack SetBounds DestroyComponents Destroying ExecuteAction FindComponent FreeNotification GetNamePath GetParentComponent HasParent InsertComponent IsImplementorOf ReferenceInterface RemoveComponent RemoveFreeNotification	Controls Handle ParentWidget Showing TabOrder TabStop Action Align Anchors BoundsRect Caption Color Constraints ControlState ControlStyle Cursor DragMode Enabled Font Height HelpContext HelpFile HelpKeyword HelpType Hint Left Name ParentFont PopupMenu ShowHint Top Width ComponentCount ComponentIndex Components ComponentState ComponentStyle DesignInfo Owner Tag
--	--	---



	SafeCallException SetSubComponent UpdateAction Assign ClassInfo ClassName ClassNamesIs ClassParent ClassType CleanupInstance DefaultHandler Dispatch FieldAddress Free FreeInstance GetInterface GetInterfaceEntry GetInterfaceTable InheritsFrom InitInstance InstanceSize MethodAddress MethodName NewInstance	
--	---	--

مدى ومجال الاستخدام Scope وأدوات التأهيل Qualifiers

مجال الاستخدام Scope يعمل على تحديد إمكانية الوصول للحقول والخصائص والأساليب الخاصة بأي كائن. وفي هذا الصدد نقول إن كافة العناصر التي يتم الإعلان عنها داخل أي كائن تكون متاحة للاستخدام لهذا الكائن وأسلافه. وبالرغم من الكود البرمجي المسئول عن تنفيذ أسلوب ما يظهر خارج منطقة الإعلان بالكائن إلا إن الأسلوب يظل داخل مدى ومجال الكائن وذلك بسبب أنه تم الإعلان عنه داخل منطقة الإعلان بالكائن.

عندما تكتب كود برمجي لتنفيذ أسلوب ما يشير إلى خصائص أو أساليب أو حقول الكائن الذي تم به الإعلان عن هذا الأسلوب في هذه الحالة لن تحتاج لأن تسبق أدوات التعريف Identifiers هذه باسم الكائن. فعلى سبيل المثال لو أنك وضعت مفتاح في فورمة جديدة فإنك تستطيع كتابة الكود البرمجي التالي لأداة معاملة الحدث OnClick المرتبط بهذا المفتاح :



```
procedure TForm1.Button1Click(Sender: TObject);
begin
    Color := clFuchsia;
    Button1.Color := clLime;
end ;
```

يمكن القول بأن أول جملة تكافئ الآتى :

```
Form1.Color := clFuchsia
```

وأنت لا تحتاج لأن تؤهل الخاصية Color بـ Form1 وذلك لأن الأسلوب Button1Click يعتبر جزء من TForm1 كما أن أدوات التعريف موجودة فى كتلة الأسلوب ومن ثم تقع داخل مدى الحالة TForm1 حيث يتم استدعاء الأسلوب.

على العكس من ذلك نجد أن الجملة الثانية تشير إلى لون كائن المفتاح (وليس إلى لون الفورمة حيث تم الإعلان عن أداة معاملة الحدث) ومن ثم فإنها تتطلب نوع من التأهيل.

تقوم لغة Delphi بإنشاء ملف لكل فورمة يكون مشتملا على الكود البرمجى الخاص بالفورمة (يسمى ملف الوحدة). ولو أنك رغبت فى الوصول لواحد من المكونات الموجودة بأى فورمة من خلال ملف الوحدة الخاص بفورمة أخرى فى هذه الحالة ستحتاج لأن تؤهل أسماء المكونات كالتالى :

```
Form2.Edit1.Color := clLime;
```

وبنفس الطريقة تستطيع أن تصل للأساليب الخاصة بأى مكون من فورمة أخرى. فعلى سبيل المثال :

```
Form2.Edit1.Clear;
```

لكى تصل إلى المكونات الموجودة فى الفورمة Form2 من ملف الوحدة الخاص بالفورمة Form1 فإنه ينبغى عليك فى هذه الحالة أيضا إضافة الوحدة الخاصة بالفورمة Form2 إلى الجملة uses الموجود فى الوحدة الخاصة بالفورمة Form1.

مدى استخدام كائن يمتد إلى الكائنات المنحدرة من هذا الكائن. وأنت تستطيع على كل حال أن تعيد الإعلان عن حقل أو خاصية أة أسلوب داخل أى من الكائنات المنحدرة. ومثل هذه العمليات (إعادة الإعلان) إما أن تكون مختفية أو متخطية override للعنصر المتوارث.

**الإعلانات الخاصة والمحمية والعامة والمنشورة**

عندما تعلن عن حقل أة خاصة أو أسلوب فى هذه الحالة تكون العضو الجديد لديه صفة الظهور والتي تحدد من خلال واحد من الكلمات المحجوزة التالية :

- o Private
- o Protected
- o Public
- o Published

صفة الظهور Visibility لأى عضو تحدد إمكانية وصوله إلى الكائنات والوحدات الأخرى.

● العضو الخاص Private يمكن الوصول إليه فقط من داخل الوحدة التي تم فيها الإعلان عن هذا العضو. وغالبا ما يتم استخدام الأعضاء الخاصة داخل قطاع وذلك للتعامل مع أساليب وخصائص أخرى (عامة أو منشورة).

● العضو المحمي protected يمكن الوصول إليه من داخل الوحدة التي فيها الإعلان عن القطاع الذى ينتمى إليه العضو كما يمكن الوصول إليه أيضا من داخل أى قطاع منحدر من هذا القطاع بغض النظر عن الوحدة الخاصة بالقطاع المنحدر.

● العضو العام Public يمكن الوصول إليه من أى موضع يمكن الوصول منه إلى الكائن الذى ينتمى إليه هذا العضو-بمعنى أنه يمكن الوصول إليه من الوحدة التي تم فيها الإعلان عن القطاع ومن أى وحدة أخرى تستخدم هذه الوحدة.

● العضو المنشور Published لديه نفس صفة الظهور التي يتمتع بها العضو العام ولكن مترجم اللغة Compiler يقوم هنا بتكوين معلومات من النوع التشغيلي runtime للأعضاء المنشورة.

الخصائص المنشورة (وهى خصائص الأعضاء المنشورة) تظهر فى النافذة Object Inspector فى مرحلة التصميم.





استخدام متغيرات الكائن Object Variables

تستطيع أن تخصص متغير كائن ما إلى متغير كائن آخر وذلك في حالة أن المتغيرات تنتمي لنفس النوع أو هناك توافق بين الأنواع التي تنتمي إليها. وإذا نظرنا للموضوع من الناحية العملية نقول أنه يمكنك تخصيص متغير كائن إلى متغير كائن آخر لو أن نوع المتغير الذي يتم التخصيص له عبارة عن جد (سلف) لنوع المتغير الذي يتم تخصيصه. فعلى سبيل المثال الكود البرمجي التالي عبارة عن إعلان عن نوع TDataForm (VCL فقط) وفي جزء الإعلان عن متغير يتم الإعلان عن المتغيرين AForm و DataForm :

type

```
TDataForm = class (TForm)
  Button1: TButton;
  Edit1: TEdit;
  DataGrid1: TDataGrid;
  Database1: TDatabase;
private
  { Private declarations }
public
  { Public declarations }
end ;
```

var

```
AForm: TForm;
DataForm: TDataForm;
```

المتغير AForm يكون من النوع TForm في حين أن المتغير DataForm يكون من النوع TDataForm. هذا وحيث إن TDataForm يعتبر خلف لـ TForm فإن جملة التخصيص التالية تعتبر صحيحة وقانونية :

```
AForm := DataForm;
```

لنفترض أنك تكتب أداة معاملة حدث للحدث OnClick لمفتاح ما بحيث عند النقر بالماوس على هذا المفتاح فإنه يتم على الفور استدعاء أداة معاملة الحدث OnClick المرتبط بهذا المفتاح. وفي هذا الصدد نقول إن كل أداة معاملة من أدوات معاملة الأحداث



تكون لديها معامل يسمى Sender الذى ينتمى للنوع TObject كما هو موضح من خلال الكود البرمجى التالى :

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  ...
  ...
  ...
end ;
```

وبسبب كون العامل Sender ينتمى إلى النوع TObject لذلك يمكن تخصيص أى كائن إلى هذا العامل. ودوما تكون قيمة العامل Sender عبارة عن أداة التحكم أو المكون الذى يستجيب للحدث. وأنت تستطيع اختبار العامل Sender لكى تعثر على نوع المكون أو أداة التحكم التى تستدعى أداة معاملة الحدث وهذا الاختبار يتم عن طريق الكلمة المحجوزة is كما هو موضح من خلال الكود البرمجى التالى :

```
if Sender is TEdit then
  DoSomething
else
  DoSomethingElse;
```

إنشاء وتدمير الكائنات

العديد من الكائنات التى تستخدمها من خلال لغة Delphi مثل المفاتيح والحقول Edit Boxes تكون ظاهرة على الشاشة سواء فى مرحلة التصميم أو فى مرحلة التشغيل Run-Time. ولكن بعض الكائنات مثل صناديق الحوار الشائعة Common dialog boxes تظهر فى مرحلة التشغيل Run-Time فقط. كما أن هناك بعض الكائنات الأخرى مثل أدوات التوقيت Timers ومكونات مصادر البيانات تكون مخفية فى مرحلة التشغيل Run-Time.

فى كثير من الأحيان ترغب فى إنشاء كائنات خاصة بك. فعلى سبيل المثال قد تقوم بإنشاء كائن TEmployee يكون مشتملا على مجموعة من الخصائص مثل الاسم Name والعنوان Title والأجر فى الساعة HourlyPayRate. وأنت تستطيع بعد ذلك



إضافة الأسلوب CalculatePay والذي يستخدم البيانات الموجودة في الخاصية HourlyPayRate لحساب مقدار ما سيتم دفعة للموظف. وفي هذا الصدد نقول إن الإعلان عن النوع TEmployee سيكون كما هو موضح من خلال الكود البرمجي التالي :

type

```
TEmployee = class (TObject)
private
    FName: string ;
    FTitle: string ;
    FHourlyPayRate: Double;
public
    property Name: string read FName write FName;
    property Title: string read FTitle write FTitle;
    property HourlyPayRate: Double read FHourlyPayRate
    write FHourlyPayRate;
    function CalculatePay: Double;
end ;
```

بالإضافة إلى الحقول والخصائص والأساليب التي قمت بتعريفها فإن TEmployee يرث كافة الأساليب الخاصة بـ TObject. وأنت تستطيع وضع الإعلان عن نوع مثل الموجود في الكود البرمجي السالف الذكر سواء في الجزء interface أو الجزء implementation بأى وحدة وبعد ذلك تنشأ حالات للقطاع الجديد عن طريق استدعاء الأسلوب Create والذي يرثه TEmployee من TObject. هذا والكود البرمجي التالي يوضح لنا كيفية القيام بذلك :

var

Employee: TEmployee;

begin

Employee := TEmployee.Create;

end ;

الأسلوب Create يطلق عليه أداة إنشاء constructor. وهو يحدد موضع بذاكرة الكمبيوتر لكائن الحالة الجديدة ويعطى إشارة مرجعية إلى الكائن.

المكونات الموجودة بأى فورمة يتم إنشاؤها وتدميرها تلقائيا بواسطة لغة Delphi. ولكن لو أنك تكتب كود برمجى خاص بك للإعلان مبدئيا عن الكائنات فى هذه الحالة ستكون مسئولاً عن عن تدمير هذه الكائنات أيضاً. وفى هذا الصدد نقول إن كل كائن يرث الأسلوب Destroy (الذى يطلق عليه أداة الهدم destructor) من TObject. على كل حال لتدمير كائن فإنه ينبغي عليك استدعاء الأسلوب Free (الذى يتم توارثه أيضا من TObject) وذلك لأن الأسلوب Free يتأكد من خلو الكائن من أى بيانات قبل أن يتم استدعاء الأسلوب Destroy. على سبيل المثال الكود البرمجى التالى:

Employee.Free

يعمل على تدمير الكائن Employee ويقوم بإخلاء الذاكرة منه.

المكونات والملكية Ownership

تمتلك لغة Delphi آلية أساسية built-in لإدارة الذاكرة. وهذه الآلية تسمح لمكون أن يفرض مسئوليته عن تفريغ مكون آخر. وفى هذه الحالة نقول أن المكون الأول يمتلك المكون الثانى. وفى هذا الصدد نقول إن الذاكرة المخصصة للمكون المملوك يتم إخلاؤها تلقائيا عندما يتم تفريغ الذاكرة المخصصة للمكون المالك.

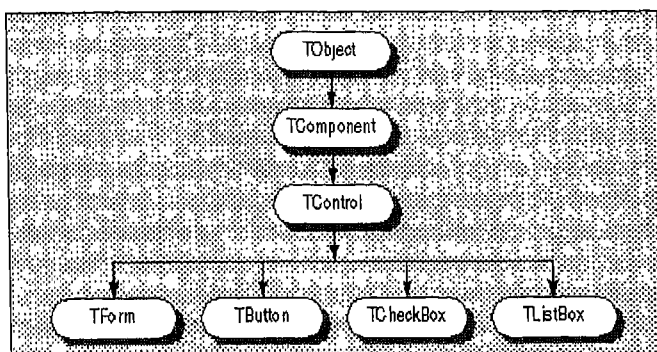
مالك أى مكون-القيمة المخصصة للخاصية Owner له-يتم تحديده عن طريق معامل يتم تمريره إلى أداة البناء فى أثناء إنشاء المكون. هذا ومن الطبيعى أن تمتلك أى فورمة كافة المكونات الموجودة بها كما إن الفورمة نفسها مملوكة للتطبيق الذى يشتمل على الفورمة. ومن ثم عند إنهاء التعامل مع التطبيق فإنه يتم على الفور وبشكل تلقائى تفريغ الذاكرة المخصصة لكافة الفورم والمكونات الموجودة بهم.

مبدأ الملكية هذا يتم تطبيقه فقط على TComponent وكافة المكونات التى تنحدر منه. فعلى سبيل لو أنك تنشأ الكائن TStringList أو الكائن TCollection (حتى ولو كان ملحقا بفورمة) فى هذه الحالة ستكون مسئولاً عن تفريغ الكائن.

لا تخلط بين مالك المكون والمكون الأب لهذا المكون.



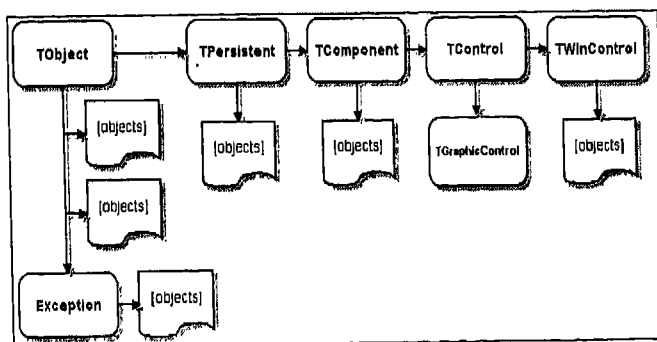
الديجرام الموضح فى الشكل التالى يعتبر مشهد مبسط جدا للهيكل الشجرى للتوارث وهو يوضح العلاقة التبادلية بين الكائنات والمكونات وأدوات التحكم :



شكل توضيحي :

كل كائن يرث من Object كما أن العديد من الكائنات ترث من TComponent. أما أدوات التحكم والتي ترث من TControl فلديها القدرة على عرض أنفسهم في مرحلة التشغيل Run-Time. فآداة تحكم مثل TCheckBox يرث كافة صفات الأداء الوظيفي الخاصة بكل من Object و TComponent و TControl كما إنه يضيف قدرات وإمكانيات خاصة لنفسه.

الديجرام التالي عبارة عن مشهد عام لمكتبة المكونات المرئية VCL (اختصار للمصطلح Visual Component Library) والتي توضح الغالبية العظمى من التفرعات الخاصة بشجرة التوارث :



شکل توضیحی :



مكتبة المكونات CLX (اختصار للمصطلح Component Library for Cross-Platform) التي أعدتها شركة Borland تشبة إلى حد كبير الديجرام الموضح في الشكل السابق ولكن الاختلاف الوحيد بينهما أن TWinControl تم استبدالها بـ TWidgetControl.



بهذا الديجران نرى العديد من قطاعات التأسيس الهامة والجدول التالي يقدم لنا

وصفا لها :

الوصف والاستخدام	القطاع
يعتبر من أهم قطاعات التأسيس على الإطلاق وهو يعتبر الجد الأعلى لأي مكون في المكتبة VCL أو المكتبة CLX. وهذا القطاع يعمل على توريث صفات السلوك الأساسية لكافة الكائنات سواء كانت VCL أو CLX وهذا التوريث يتم عن طريق مجموعة من الأساليب التي تؤدي الوظائف الأساسية مثل إنشاء وصيانة وتدمير أي حالة لأي كائن.	TObject
يعتبر قطاع التأسيس بكافة القطاعات التي ترتبط بالإستثناءات. والقطاع التأسيس هذا يعمل على توفير وسيط محكم ومتكامل لحالات الأخطاء كما يعمل أيضا على تمكين التطبيقات من التعامل مع حالات الخطأ بطريقة رائعة وفعالة.	Exception
قطاع التأسيس هذا مخصص لكافة الكائنات التي تنفذ الخصائص. هذا والقطاعات التي توجد تحت قطاع التأسيس TPersistent تتعامل مع إرسال البيانات إلى الـ Streams وتسمح بتخصيص القطاعات.	TPersistent
قطاع التأسيس هذا مخصص لكافة المكونات الغير مرئية مثل المكون TApplication. وهذا القطاع يعتبر الجد المشترك لكافة المكونات.	TComponent



<p>وهو يسمح لأي مكون بأن يتم عرضه بباليتة المكونات كما يسمح أيضا للمكون بأن يمتلك مكونات أخرى وكذلك يسمح للمكون بأن يتم التعامل معه مباشرة داخل أي فورمة.</p>	
<p>عذا القطاع يمثل قطاع التأسيس لكافة أدوات التحكم التي تكون ظاهرة في مرحلة التشغيل Run-Time. وهو يعتبر الجد المشترك لكافة المكونات المرئية ويعمل على توفير أدوات تحكم مرئية قياسية كثل Position و cursor كما يعمل أيضا على توفير أحداث تستجيب لأفعال الماوس.</p>	TControl
<p>قطاع التأسيس هذا مخصص لكافة الكائنات التي تستخدم في تصميم واجهات الاستخدام UI وهو يطلق عليه أيضا Widgets. وفي هذا الصدد نقول إن أدوات التحكم الموجودة تحت قطاع التأسيس هذا يقال عليها أنها windowed controls ولديها القدرة على التقاط مدخلات لوحة المفاتيح. (في مكتبة المكونات CLX نجد TWidgetControl بدلا من TWinControl).</p>	TWinControl

فرع قطاع التأسيس TObject

فرع قطاع التأسيس TObject يشتمل على كافة الكائنات التي تنحدر من TObject وفي نفس الوقت لا يشتمل على الكائنات التي تنحدر من TPersistent. وفي هذا الصدد نقول إن كافة الكائنات الموجودة في VCL أو في CLX تنحدر من TObject عبارة عن قطاع استخلاص يمتلك عدد من الطرق التي تعمل على تعريف الصفات الأساسية للسلوك مثل الإنشاء والتدمير والرسائل أو تعامل النظام مع الأحداث. وأغلب القوة التي تتمتع بها الكائنات VCL و CLX يتم الحصول عليها من خلال الأساليب التي يقدمها TObject.

قطاع التأسيس TObject يعمل على توريث صفات السلوك الأساسية لكافة الكائنات في VCL و CLX وذلك عن طريق توفير أساليب تعمل على توفير الآتي :

● القدرة على الاستجابة عندما يتم إنشاء حالات للكائن أو عندما يتم تدمير أى من هذه الحالات.

● النوع Class ومعلومات الحالة بأى كائن ومعلومات نوع مرحلة التشغيل Run-Time (RTTI) حول الخصائص المنشورة الخاصة به.

● الدعم الخاص بإمكانية التعامل مع الرسائل (بالنسبة لكائنات المكتبة VCL فقط).
قطاع التأسيس TObject يعتبر الجذع المباشر للعديد من القطاعات البسيطة. فالقطاعات التى تكون موجودة داخل هذا الفرع تشترك جميعها فى صفة واحدة وهامة جدا : فكلهم مؤقتون سريعوا الزوال. وهذا يعنى أن هذه القطاعات لا تمتلك أى أسلوب لحفظ الحالة التى يكونوا عليها قبل أن يتم تدميرهم ومن ثم يمكن القول بأنهم غير دائمين.

واحد من المجموعات الأساسية للقطاعات فى هذا الفرع هى القطاع Exception. فهذا القطاع يعمل على توفير مجموعة هائلة من قطاعات الاستثناء الأساسية Built-In والتى يتم من خلالها التعامل تلقائيا مع الأخطاء التى تنتج عن القسمة على أصفار وأخطاء التعامل مع ملفات I/O وأخطاء تفصيل أنواع خاصة غير صحيحة invalid typecasts بالإضافة إلى العديد من حالات الأخطاء الأخرى.

هناك نوع آخر من المجموعات فى الفرع TObject عبارة عن القطاعات التى تغلف هياكل البيانات مثل :

● القطاع TBits الذى يعمل على تخزين مصفوفة من القيم المنطقية Boolean.

● القطاع TList الذى يضم قائمة بالقطاعات المرتبطة معا.

● القطاع TStack الذى يضم مجموعة من المؤشرات last-in first-out.

● القطاع TQueue الذى يضم مجموعة من المؤشرات first-in first-out.

فى المكتبة VCL تستطيع أيضا العثور على wrappers للكائنات الخارجية مثل TPrinter (الذى يغلف واجهة الاستخدام التى يتم من خلالها التعامل مع الطابعة المثبتة ببيئة الويندوز) و TRegistry الذى يعد Wrapper منخفضة المستوى لأحقية استخدام النظام والدوال التى تتعامل مع أحقية الاستخدام registry.



مثل هذه العناصر السالفة الذكر مخصصة للتطبيقات التى يتم تصميمها للعمل ببيئة الويندوز فقط.



TStream يعتبر مثال جيد لنوع آخر من القطاعات الموجودة فى هذا الفرع. وفى هذا الصدد نقول إن TStream عبارة عن نوع قطاع التأسيس لكائنات التدفق Stream Objects والتي يمكن الحصول منها على معلومات (أو تخصيص لها بيانات) من خلال أنواع مختلفة من وسائط التخزين مثل الاسطوانات التخزين (الصلبة أو المرنة) والذاكرة الديناميكية.

لقد رأيت أن هذا الفرع يضم العديد من القطاعات المتعددة الأنواع والتي تعتبر مهمة جدا بالنسبة لك فى أثناء إعداد وتطوير البرامج من خلال لغة Delphi.

فرع قطاع التأسيس TPersistent

الكائنات الموجودة فى هذا الفرع من المكتبة VCL والمكتبة CLX تنحدر من قطاع التأسيس TPersistent الذى يعمل على إضافة صفة الاستمرار persistence. وهذه الصفة تحدد ما الذى يتم حفظه مع ملف الفورمة أو Module البيانات وما الذى سيتم تحميله داخل الفورمة أو Module البيانات عندما يتم استرجاع أى منهما من الذاكرة.

الكائنات الموجودة فى هذا الفرع تقدم عدد من الخصائص للمكونات. وهذه الخصائص هى الوحيدة التى يتم تحميلها وحفظها مع أى فورمة وذلك لو أن لديهم مالك. وفى هذا الصدد نقول إن المالك ينبغى أن يكون مكون. وهذا الفرع يقدم الدالة GetOwner التى تسمح لك بأن تحدد المالك لأى خاصية.

الكائنات الموجودة فى هذا الفرع تعتبر أولى الكائنات التى تشتمل على جزء النشر والذى يمكن فيه تحميل وحفظ الخصائص تلقائيا. كذلك فإن الأسلوب DefineProperties يسمح لك بأن تحدد كيفية تحميل وحفظ الخصائص.

فيما يلى سنستعرض سويا بعض من القطاعات الأخرى الموجودة فى الفرع

: TPersistent

- القطع TGraphicsObject وهو عبارة عن قطاع تأسيس مستخلص للكائنات الرسومية (التي تستخدم في الرسم) مثل الكائن TBrush والكائن TFont والكائن TPen.
- القطع TGraphic وهو عبارة عن قطاع تأسيس مستخلص لكائنات مثل الأيقونات والصور النقطية bitmaps والتي يمكنها تخزين وعرض صور مرئية : TBitmap و TIcon (و TMetafile المخصصة لبيئة الويندوز فقط).
- القطع TStrings وهو عبارة عن قطاع تأسيس مستخلص للكائنات التي تمثل مجموعة من السلاسل الحرفية.
- القطع TClipboard الذي يشتمل على النصوص أو العناصر الرسومية التي تم قصها أو نسخها من أى تطبيق.
- كل من القطع TCollection والقطع TOwnedCollection والقطع TCollectionItem. وهذه القطاعات تستخدم للاحتفاظ بمجموعات مفهرسة من العناصر الخاصة.

فرع قطاع التأسيس TComponent

- فرع قطاع التأسيس TComponent يشتمل على الكائنات التي تنحدر من TComponent وفي نفس الوقت لا تنحدر من TControl. والكائنات الموجودة في هذا الفرع عبارة عن مكونات تستطيع التعامل معها بالفورم في مرحلة التصميم. وهي تعتبر كائنات لها صفة الاستمرار ولديها القدرة على القيام بالآتي :
- الظهور في باليئة المكونات مع إمكانية تغييرها في أداة تصميم الفورمة.
- امتلاك وإدارة مكونات أخرى.
- تحميل وحفظ أنفسها.

هناك العديد من الأساليب في TComponent تعمل على تحديد طريقة عمل المكونات في مرحلة التصميم كما تحدد أيضا المعلومات التي سيتم حفظها مع كل مكون. ونود هنا القول بأن مبدأ الـ Streaming تم التقديم له في هذا الفرع بكل من المكتبة VCL



والمكتبة CLX. وفى هذا الصدد نقول إن لغة Delphi تقوم بتنفيذ أغلب العمليات التى تعتمد على مبدأ الـ Streaming. أما بالنسبة للخصائص فىكون لها صفة الوجود والاستمرار فى حالة أنه قد تم نشرهم ونود هنا القول بأن الخصائص المنشورة تكون Streamed تلقائيا. بالإضافة لما سبق نجد أن قطاع التأسيس TComponent يعمل أيضا على تقديم المفهوم الأساسى لمبدأ الملكية Ownership الشائع بين الكائنات الموجودة بكل من المكتبة VCL والمكتبة CLX. وفى هذا الصدد نقول إن هناك خاصيتين تعملان على تدعيم مبدأ الملكية وهما الخاصية Owner والخاصية Components. هذا وكل مكون لديه الخاصية Owner التى تشير إلى مكون آخر على أساس أنه المالك له. فالمكون يمكن أن يمتلك مكونات أخرى. وفى هذه الحالة يتم الإشارة مرجعيا إلى كافة المكونات المملوكة فى الخاصية Array للمكون المالك.

أداة البناء الخاصة بالمكون تتعامل مع معامل واحد فقط يتم استخدامه لتحديد المالك الجديد للمكون. ولو أن المالك موجود بالفعل فى هذه الحالة يتم إضافة مكون جديد إلى قائمة المكونات المملوكة للمالك. هذا وبعيدا عن استخدام قائمة المكونات للإشارة مرجعيا إلى المكونات المملوكة فإن هذه الخاصية يمكن استخدامها أيضا لعمل تدمير تلقائى للمكونات المملوكة. وطالما أن المكون لديه مكون فإنه فى هذه الحالة يتم تدمير هذا المكون عندما يتم تدمير المالك له. فعلى سبيل المثال حيث إن TForm ينحدر من TComponent لذلك فإن كافة المكونات المملوكة للفورمة يتم تدميرها وإخلاء مواقعها بالذاكرة عندما يتم تدمير الفورمة. وهذا المبدأ يفترض أن كافة المكونات الموجودة بالفورمة تعمل على تنظيف نفسها بطريقة جيدة عندما يتم استدعاء أدوات الهدم الخاصة بهم.

لو أن نوع الخاصية كان TComponent أو أحد المكونات التى تنحدر منه فى هذه الحالة يقوم نظام الـ Streaming بإنشاء حالة لهذا النوع عندما يتم قراءته. أما لو كان نوع الخاصية عبارة عن TPersistent وفى نفس الوقت ليس TComponent فإن نظام الـ Streaming يستخدم الحالة الموجودة حاليا والمتاحة من خلال الخاصية ويقوم بقراءة القيم الخاصة بخصائص هذه الحالة.

عند إنشاء ملف فورمة (ملف يتم استخدامه لتخزين معلومات عن المكونات الموجودة فى الفورمة) فإن أداة تصميم الفورمة تدور عبر مصفوفة المكونات الخاصة به ويحفظ كافة المكونات الموجودة فى الفورمة. وكل مكون يكون لديه علم عن كيفية كتابة خصائصه المتغيره إلى Stream (فى هذه الحالة يكون الـ Stream عبارة عن ملف نصي). وعلى النقيض عند تحميل خصائص المكونات فى ملف الفورمة فإن أداة تصميم الفورمة يدور داخل مصفوفة المكونات ويقوم بتحميل كل مكون على حدة.

أنواع تقاطعات التى ستجدها فى هذا الفرع تتضمن الآتى :

● القطع TMainMenu وهو يعمل على توفير شريط قائمة والقوائم المنسدلة منه drop-down وذلك داخل أى فورمة.

● القطع TTimer وهو يتضمن دوال التوقيت.

● كل من القطع TOpenDialog والقطع TSaveDialog والقطع TFontDialog والقطع TColorDialog وغيرهم من القطاعات الأخرى تعمل على توفير صناديق الحوار الشائعة الاستخدام.

● القطع TActionList الذى يحتفظ بقائمة تتضمن الأفعال المستخدمة مع المكونات وأدوات التحكم مثل العناصر الموجودة بالقوائم والمفاتيح.

● القطع TScreen والذى يتتبع الفورم وModules البيانات التى تم إعداد حالات ابتدائية لها بواسطة التطبيق كما يحدد الفورمة النشطة حاليا كما يعمل أيضا على نقل دفة التحكم داخل الفورمة النشطة حاليا كما يتتبع أيضا كل من حجم ودرجة وضوح الشاشة والمؤشرات (سواء الكتابة أو الماوس) بالإضافة إلى الفوننت المتاحة للتطبيق لكى يستخدمها.

المكونات التى لا تحتاج لواجهة استخدام مرئية يمكن الحصول عليها مباشرة من TComponent. هذا ولكى تصنع أداة مثل معدة التوقيت TTimer فإنك تستطيع فى هذه الحالة أن تنشأ هذه الأداة مباشرة من TComponent. وهذا النوع من المكونات يوجد فى بالليته المكونات ولكنه يؤدى وظائف داخلية يمكن التحكم فيها من خلال الكود البرمجى وذلك بدلا من الظهور فى واجهة الاستخدام فى مرحلة التشغيل Run-Time.

فى المكتبة CLX نجد أن الفرع TComponent يشتمل أيضا على THandleComponent. وهو عبارة عن قطاع تأسيس خاص بالمكونات الغير مرئية التى تتطلب نوع من المعاملة فى أثناء مرحلة التشغيل Run-Time مثل صناديق الحوار والقوائم.



فرع قطاع التأسيس TControl

فرع قطاع التأسيس TControl يتألف من المكونات التى تنحدر من TControl وليس من TWinControl (قطاع التأسيس TWidgetControl فى المكتبة CLX). والكائنات الموجودة فى هذا الفرع عبارة عن أدوات تحكم هى فى حد ذاتها كائنات مرئية بمعنى أن مستخدم التطبيق يستطيع رؤيتها والتعامل معها فى مرحلة التشغيل Run-Time.

هذا وكافة أدوات التحكم لديها خصائص وأساليب وأحداث تتحكم فى مظهر أدوات التحكم على الشاشة مثل موضعه والمؤشر المرتبط بنافذة أداة التحكم (أو ال Widget فى المكتبة CLX) وطرق رسم أو نقل أداة التحكم والأحداث التى تقع عندما تستجيب أداة التحكم لأفعال الماوس.

أدوات التحكم لا تستقبل أى مدخلات من لوحة المفاتيح.



فى حين أن TComponent يعمل على تعريف وتحديد سلوك كافة المكونات التى تتبعه نجد أن TControl يعمل على تعريف وتحديد سلوك كافة أدوات التحكم المرئية فقط. وهذا يتضمن روتينات الرسم والأحداث القياسية والملكية.

هناك نوعين أساسيين من أدوات التحكم هما :

- أدوات تحكم لديها نافذة خاصة بها.
- أدوات تحكم تستخدم نافذة أدوات التحكم التى تكثل الأب لها.

أدوات التحكم التي لديها نافذة خاصة بها يطلق عليها Windowed Controls (في المكتبة VCL) أو يطلق عليها Widget-based Controls (في المكتبة CLX) وهي تنحدر من TWinControl (بالمكتبة VCL) أو من TWidgetControl (بالمكتبة CLX).

كل من المفاتيح ومربعات الاختبار تنتمي لهذه النوعية من أدوات التحكم.



أما أدوات التحكم التي تستخدم النافذة الخاصة بأداة التحكم الأب فيطلق عليها أدوات تحكم رسومية Graphic وهي تنحدر من TGraphicControl.

كل من أداة التحكم Image وأداة التحكم Label تنتمي إلى هذه النوعية من أدوات التحكم.







في المكتبة VCL نجد أن الاختلاف الأساسي بين هذه الأنواع من المكونات يتمثل في أن أدوات التحكم الرسومية لا تمتلك نافذة معاملة ومن ثم فهي لا تستطيع استقبال دفة التحكم لإدخال البيانات.

أما في المكتبة CLX نجد أن الاختلاف الأساسي بين هذه الأنواع من المكونات يتمثل في أن أدوات التحكم الرسومية لا تمتلك Widget خاصة بها ومن ثم لا تتمكن من استقبال دفة التحكم لإدخال البيانات كما إنها لا تستطيع أيضا احتواء أدوات تحكم أخرى داخلها.

بسبب أن أدوات التحكم الرسومية لا تحتاج لمعاملة لذلك فإن متطلباتها من مصادر نظام التشغيل تكون في أدنى مستوى كما أن رسم أي أداة من أدوات التحكم الرسومية يكون أسرع من رسم نفس النوع من أدوات التحكم بالمكتبة CLX.

أدوات التحكم التي تنحدر من TGraphicControl يجب أن ترسم نفسها بالفورمة وهي تتضمن أدوات التحكم التالية :

أداة التحكم	الأيقونة	الصفحة في البيئة المكونات	الوصف والاستخدام
TImage		Additional	عرض الصور الرسومية.
TLabel		Standard	عرض نص بالفورمة.
TBevel		Additional	تمثيل التخطيط البارز أو المحفور.
TPaintBox		System	توفير نسيج يمكن للتطبيقات استخدامه لرسم صور أو إعادة تكوينها Rendering.

لاحظ أن أدوات التحكم هذه تضم نفس روتينات الرسم (مثل Repaint و Invalidate وغيرها) ومن ثم فهي لا تحتاج أبدا لاستقبال دقة التحكم في أثناء مرحلة التشغيل Run-Time.



فرع قطاع التأسيس TWinControl

فرع قطاع التأسيس TWinControl (وقطاع التأسيس TWidgetControl في المكتبة CLX) يتضمن كافة أدوات التحكم التي تنحدر من TWinControl الذي يعتبر قطاع التأسيس لكافة أدوات التحكم التي لها نافذة بما فيها العديد من العناصر التي ستستخدمها في أثناء إعداد واجهة الاستخدام للتطبيق الذي تعده.

أما بالنسبة لـ TWidgetControl فهو يعتبر قطاع التأسيس لكافة أدوات التحكم التي لها نافذة والموجودة بالمكتبة CLX. وهي تستخدم أيضا في إعداد واجهات الاستخدام للتطبيقات. والأمثلة على هذه النوعية من أدوات التحكم كثيرة ومنها المفاتيح والعناوين وشرائط الحركة Scroll bars سواء الأفقية أو الرأسية.

فيما يلي المظاهر والإمكانات التي تتمتع بها أدوات التحكم التي لها نوافذ سواء التي توجد في المكتبة VCL أو في المكتبة CLX :

- هذه النوعية من أدوات التحكم يمكنها استقبال دفة التحكم في أثناء التعامل مع التطبيق في مرحلة التشغيل Run-Time.
 - أدوات التحكم الأخرى قد تعرض بيانات ولكن المستخدم لا يتمكن من استخدام لوحة المفاتيح للتفاعل مع هذه النوعية من أدوات التحكم.
 - يمكن لهذه النوعية من أدوات التحكم اشتغال أدوات تحكم أخرى داخلها.
 - أداة التحكم التي تشتمل على أدوات تحكم أخرى يطلق عليها الأب Parent أو الأصل وفي هذا الصدد نقول أن هذه النوعية من أدوات التحكم يمكن أن تكون أب لأداة تحكم واحدة أو أكثر (يطلق عليها أدوات التحكم الوليدة Child أو التابعة).
 - هذه النوعية من أدوات التحكم لديها ما يعرف نافذة المعاملة window handle.
- الكائنات التي تنحدر من TWinControl (أو من TWidgetControl بالمكتبة CLX) عبارة عن أدوات تحكم تستطيع استقبال دفة التحكم مما يعنى أن لديها القدرة على استقبال المدخلات من مستخدم التطبيق من خلال لوحة المفاتيح. وهذا يؤكد على أن أدوات التحكم هذه لديها عدد أكبر من الأحداث.
- هذا الفرع يتضمن كل من أدوات التحكم التي يتم رسمها تلقائياً (مثل TEdit و TListBox و TComboBox و TPageControl وغيرهم كثير...) وأدوات التحكم المفصلة التي يجب على لغة Delphi رسمها (مثل TDBNavigator و TMediaPlayer) بالمكتبة VCL فقط) و TGUIage (بالمكتبة VCL فقط) وغيرهم كثير...).
- الكائنات التي تنحدر مباشرة من TWinControl (أو من TWidgetControl بالمكتبة CLX) تعتبر أدوات تحكم قياسية مثل الحقول والقوائم المنسدلة وقوائم العرض و Page Control ومن ثم لديها القدرة على رسم نفسها بالفورمة.
- القطاع TCustomControl يتم توفيره من أجل المكونات التي تتطلب نافذة معاملة ولكن لا تشتمل على أداة من أدوات التحكم القياسية والتي تتضمن القدرة على إعادة رسم نفسها بنفسها. هذا وينبغي عليك ألا تقلق أبداً بخصوص الطريقة التي تقوم بها أدوات التحكم لإعادة تكوين نفسها أو بخصوص طريقة استجابتهم للأحداث التي تقع لهم حيث إن لغة Delphi هي التي تتولى بالكامل تنفيذ مثل هذه العمليات بدلا منك.



الخصائص المشتركة لأدوات التحكم المنحدرة من قطاع التأسيس TControl

كافة أدوات التحكم المرئية (التي تنحدر من قطاع التأسيس TControl) تتشارك معا في خصائص معينة من بينها ما يلي :

- خصائص الأفعال Action Properties.
 - الخصائص التي تحدد الموضع والحجم والضبط Alignment.
 - الخصائص التي تحدد طريقة العرض.
 - خصائص الأب Parent Properties.
 - خاصية التجول Navigation Property.
 - خصائص السحب والإسقاط Drag-and-Drop.
 - خصائص السحب والإرساء Drag-and-Dock (للكائنات الموجودة بالمكتبة VCL فقط).
- في حين أن هذه الخصائص يتم توارثها من قطاع التأسيس TControl إلا إنه يمكن نشرها- من ثم تظهر في النافذة Object Inspector- فقط للمكونات التي يمكن تطبيق هذه الخصائص عليها. فعلى سبيل المثال نقول أن TImage لا يعمل على نشر الخاصية Color الخاصة به ومن ثم فإنه يتم تحديد لونه عن طريق الرسمة التي يتم عرضها داخله.

خصائص الأفعال Action Properties

الأفعال تسمح لك بأن تستخدم نفس الكود البرمجي لأداء العديد من الأفعال (فعلى سبيل المثال عندما يقوم كل من أيقونة في شريط أدوات وعنصر بقائمة بعمل نفس الشيء) بالإضافة إلى توفير طريقة واحدة ومركزية لتنفيذ الأفعال أو تعطيلها وذلك بناء على الحالة التي يكون عليها التطبيق في مرحلة التشغيل Run-Time.

الخاصية	الوصف والاستخدام
Action	تمييز وتحديد الفعل المرتبط بأداة التحكم.
ActionLink	هذه الخاصية تشتمل على كائن ربط الفعل الملحق بأداة التحكم.

خصائص تحديد الموضع والحجم والضغط Alignment

هذه المجموعة من الخصائص تعمل على تحديد كل من موضع وحجم أداة التحكم داخل أداة التحكم الأصلية أو الأب :

الخاصية	الموقع والاستخدام
Height	تحديد ارتفاع (الحجم الرأسى) أداة التحكم.
Width	تحديد عرض (الحجم الأفقى) أداة التحكم.
Top	تحديد موضع الحافة العليا لأداة التحكم.
Left	تحديد موضع الحافة اليسرى لأداة التحكم.
AutoSize	تقرير ما إذا كانت أداة التحكم تقوم بتغيير حجمها تلقائياً بحيث يصبح حجمها مناسب لمحتوايتها.
Align	تحديد الجانب الذى ستوجد به أداة التحكم داخل أداة التحكم الـ Parent التى توجد بها (اليمين أو اليسار أو فى الوسط).
Anchor	تحديد طريقة تعليق أداة التحكم داخل أداة التحكم الـ Parent التى توجد بها.

الخاصية Anchor تكون للكائنات الموجودة فى المكتبة VCL فقط.



المجموعة التالية من الخصائص تعمل على تحديد كل من ارتفاع وعرض والحجم الكلى ل منطقة الـ Client الخاصة بأداة التحكم :

الخاصية	الموقع والاستخدام
ClientHeight	تحديد ارتفاع منطقة الـ Client الخاصة بأداة التحكم وهذا الارتفاع يقاس بالبكسل.
ClientWidth	تحديد عرض منطقة الـ Client الخاصة بأداة التحكم وهذا العرض يقاس بالبكسل.



هذه الخصائص لا يمكن الوصول إليها في المكونات الغير مرئية ولكن على كل حال تعمل لغة Delphi على تتبع الموضع الذى تضع فيه أيقونات المكونات بالفورم التى تعدها. ونود هنا القول بأنه فى أغلب الأحيان ستقوم بتحديد قيم هذه الخصائص عن طريق التعامل مع صورة أداة التحكم الموجودة فى الفورمة أو عن طريق استخدام باليتة الضبط .Alignment Palette

خصائص العرض Display Properties

فيما يلى مجموعة الخصائص التى تتحكم فى المظهر العام لأى أداة تحكم :

الخاصية	الوصف والاستخدام
Color	تغيير لون الخلفية لأداة التحكم.
Font	تغيير كل من لون ونوع ونمط وحجم الحروف.
Cursor	تحديد الصورة المستخدمة لتمثيل مؤشر الماوس عندما يمر فوق المنطقة التى تحتلها أداة التحكم بالفورمة.
DesktopFont	تحديد ما إذا كانت أداة التحكم تستخدم فونتيات أيقونات الويندوز عند كتابة النصوص.

الخاصية DesktopFont تكون لأدوات التحكم الموجودة فى المكتبة VCL فقط.



خصائص الأب Parent Properties

لكى تعمل على تحسين وتطوير المظهر العام للتطبيق الذى تتولى إعداده فإنك تستطيع إذن جعل أى أداة تحكم تشبه الحاوية التى يوجد بداخلها- هذه الحاوية يطلق عليها الأب Parent- وذلك عن طريق تخصيص القيمة المنطقية True للخصائص التالية والتى تخص الحاوية (الأب) :

الخاصية	الوصف والاستخدام
ParentColor	تحديد أين تبحث أداة التحكم عن معلومات اللون الخاصة بها.
ParentFont	تحديد أين تبحث أداة التحكم عن معلومات الفونت الخاصة بها.
ParentShowHint	تحديد أين تبحث أداة التحكم لكي تعثر على معلومات المساعدة الخاصة بها.

خاصية التجول Navigation Property

الخاصية التالية تعمل على تحديد الطريقة التي يتبعها المستخدمون للتطبيق للتجول بين أدوات التحكم الموجودة في أى فورمة :

الخاصية	الوصف والاستخدام
Caption	هذه الخاصية تشتمل على سلسلة حرفية تعتبر عنوان label للمكون (أداة التحكم). هذا ولكي تجعل أى حرف فى هذه السلسلة الحرفية تحته خط (وفى هذه الحالة يعرف هذا الحرف بحرف الاستدعاء) ضع العلامة & قبل الحرف مباشرة. ومن ثم يستطيع المستخدم فى مرحلة التشغيل Run-Time التعامل مع المكون أو أداة التحكم عن طريق الضغط على هذا الحرف مع المفتاح Alt فى نفس الوقت.

خصائص السحب والإسقاط Drag-and-Drop

الخصائص التالية تؤثر بشكل مباشر على الطريقة التي تتم بها عملية السحب والإسقاط لأدوات التحكم :

الخاصية	الوصف والاستخدام
DragMode	هذه الخاصية تحدد الطريقة التي تبدأ بها عملية السحب. والقيمة الافتراضية لهذه الخاصية عبارة عن dmManual كما أن التطبيق ينبغي عليه استدعاء الأسلوب BeginDrag للبدء فى عملية



السحب. أما عندما تكون قيمة هذه الخاصية عبارة عن dmAutomatic فى هذه الحالة تبدأ عملية السحب بمجرد أن الضغط على زر الماوس.	
تعمل هذه الخاصية على تحديد شكل مؤشر الماوس عندما يمر فوق المكونات أو أدوات التحكم التى تقبل أن تجرى معها عملية السحب والاسقاط.	DragCursor

الخاصية DragCursor تكون لأدوات التحكم الموجودة فى المكتبة VCL فقط.



خصائص السحب والإرساء Drag-and-Dock

فيما يلى نقدم لك مجموعة الخصائص التى تتحكم فى سلوك عملية السحب والإرساء Drag-and-Dock التى تجرى على أدوات التحكم التى تنحدر من TWinControl :

الخاصية	الوصف والاستخدام
Floating	تحديد ما إذا كانت أداة التحكم عائمة أم لا.
DragKind	تحديد ما إذا كانت أداة التحكم قد تم سحبها بطريقة عادية أم تم سحبها لإرسائها بأحد جوانب الشاشة.
DragMode	تحديد طريقة البدء فى إجراء عملية السحب والإسقاط أو عملية السحب والإرساء على أداة التحكم.
FloatingDockSiteClass	تحديد قطاع أداة التحكم المؤقتة الذى يستضيف أداة التحكم عندما تكون عائمة وسط الشاشة.
DragCursor	عبارة عن شكل مؤشر الماوس الذى يظهر على الشاشة فى أثناء عملية السحب.

DockOrientation	تحديد طريقة ارساء أداة التحكم بالنسبة لأدوات التحكم الأخرى التى ترسو الآن فى الحاوية (الأب).
HostDockSite	تحديد أداة التحكم التى سيتم الإرساء بها.
LRDockWidth	تحديد عرض أداة التحكم عندما ترسو أفقيا.
TBDockHeight	تحديد ارتفاع أداة التحكم عندما ترسو رأسيا.
UnDockHeight	تحديد ارتفاع أداة التحكم عندما تكون عائمة وسط أداة التحكم الحاوية.
UnDockWidth	تحديد عرض أداة التحكم عندما تكون عائمة وسط أداة التحكم الحاوية.

هذه الخصائص السالفة الذكر تكون لأدوات التحكم الموجودة فى المكتبة VCL فقط.



الفصل الرابع

تمرين عملي
إنشاء تطبيق بسيط
لتحرير النصوص



مقدمة عامة

من خلال هذا التمرين العملي سنتعرف سويا على كيفية إنشاء محرر للنصوص متكامل يشتمل على مجموعة من الفوائم وشريط أدوات وشريط حالة.

هذا التمرين العملي يمكن تنفيذه من خلال كافة إصدارات لغة Delphi كما أن التطبيق الناتج سيكون قادر على العمل من خلال بيئة الويندوز فقط.



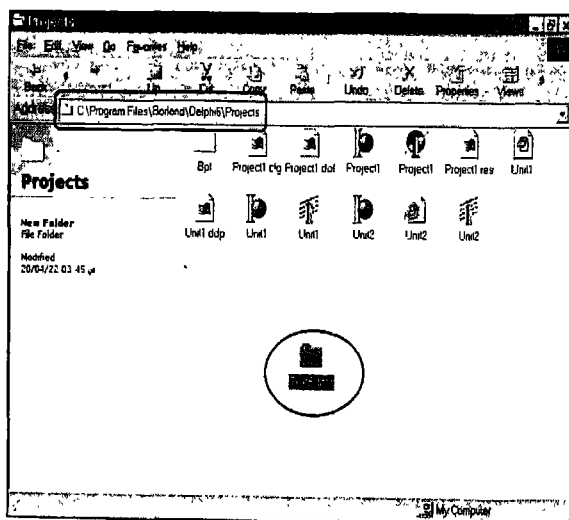
البدء في تطبيق جديد

قبل في تبدأ في تطبيق جديد قم بإنشاء الفهرس الذي سيتم فيه تخزين ملفات التطبيق التي سيتم إنشاؤها من خلال Delphi وللقيام بذلك اتبع الخطوات التالية :

(١) قم بإنشاء فهرس يسمى TextEditor وذلك بالمسار التالي :

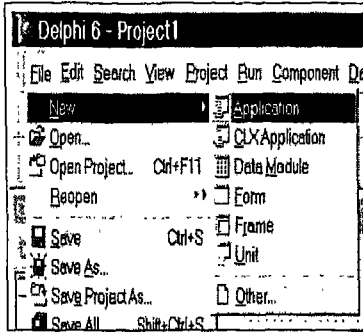
C:\Program Files\Borland\Delphi6\Projects

كما هو موضح بالشكل التالي :



شكل توضيحي :

(٢) افتح مشروع جديد بلغة Delphi وذلك من عن طريق فتح القائمة File ثم اختيار New ثم اختيار Application كما هو موضح بالشكل التالي :



شكل توضيحي :

كل تطبيق يتم التعامل معه على أساس أنه مشروع. وعندما تبدأ في التعامل مع لغة Delphi فإنها تقوم بشكل افتراضي بإنشاء مشروع فارغ.



لو كان هناك مشروع آخر مفتوح بالفعل في هذه الحالة افتح القائمة File ثم اختر منها New ثم اختر Application كما ذكرنا سابقا.



عندما تفتح مشروع جديد تقوم لغة Delphi على الفور وبشكل تلقائي بإنشاء

الملفات التالية :

الملف Project1.dpr وهو عبارة عن ملف كود مصدري يكون ملحقا بالمشروع. وهذا الملف يطلق عليه ملف المشروع.

الملف Unit1.pas وهو عبارة عن ملف كود مصدري يكون ملحقا بفورمة المشروع الأساسية. وهذا الملف يطلق عليه ملف الوحدة.

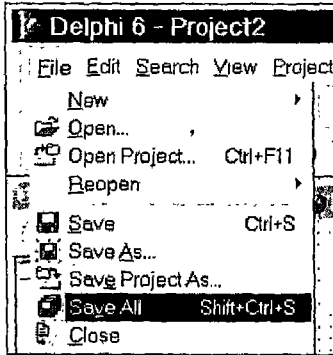
الملف Unit1.dfm وهو عبارة عن ملف مصدري يعمل على تخزين معلومات عن فورمة المشروع الأساسية. وهو يطلق عليه ملف الفورمة.

كل فورمة لديها كل من ملف الوحدة (Unit1.pas) وملف الفورمة (Unit1.dfm) الخاص بها. ولو أنك قمت بإنشاء فورمة ثانية سيكون لها كل من ملف الوحدة (Unit2.pas) وملف الفورمة (Unit2.dfm) الخاص بها وهكذا... ومثل هذه الملفات يتم إنشاؤها تلقائيا.



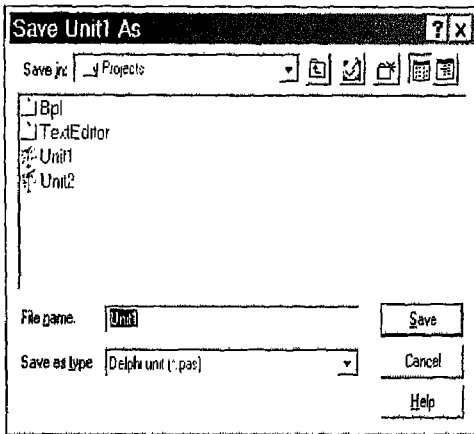


(٣) افتح القائمة File ثم اختر منها الأمر Save All كما هو موضح بالشكل التالي وذلك لكي تحفظ الملفات الخاصة بالمشروع الذي تعده :



شكل توضيحي :

(٤) والآن يظهر على الشاشة صندوق الحوار Save Unit1 As الموضح في الشكل التالي :



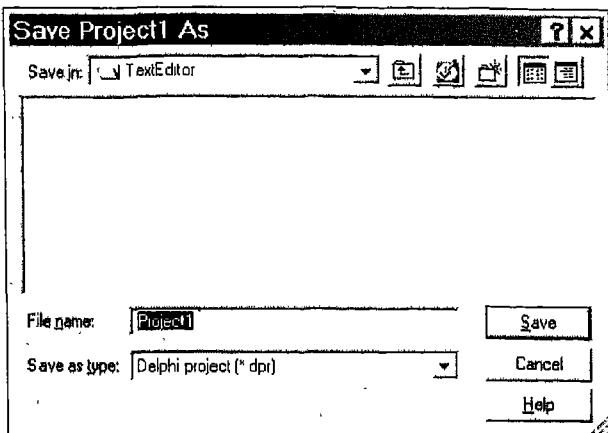
شكل توضيحي :

من خلال صندوق الحوار هذا قم بالآتي :

➤ الذهاب إلى المجلد TextEditor الذي قمت بإنشاؤه.

➤ احفظ الوحدة Unit1 مستخدماً الاسم الافتراضي لها وهو Unit1.pas.

(٥) بعد ذلك يظهر على الشاشة صندوق الحوار Save Project1 As الموضح في الشكل التالي :



شكل توضيحي :

من خلال صندوق الحوار هذا قم بالآتي :

● حفظ المشروع مستخدماً الاسم (TextEditor.dpr).

الملف التنفيذي لهذا المشروع سيكون بنفس اسم المشروع ولكن بالامتداد .exe



فيما بعد ستتمكن من إعادة حفظ ما تقوم به من عمل عن طريق الأمر Save All بالقائمة File ولكن في هذه المرة لن تظهر صناديق الحوار السالفة الذكر.



عندما تحفظ المشروع الذي تعده فإن Delphi تقوم بإنشاء عدد من الملفات الإضافية وتضعها في الفهرس الذي أعدته للمشروع (الفهرس TextEditor). ومن بين هذه الملفات كل من الملف TextEditor.dof وهو عبارة عن ملف خيارات الديلفي والملف TextEditor.cfg وهو عبارة عن ملف التهيئة والملف TextEditor.res وهو ملف مصادر الويندوز.

لست في حاجة لأن تقلق بخصوص هذه الملفات ولكن لا تحاول مسح هذه الملفات فهي ضرورية بالنسبة للمشروع الذي تتولى إعداده.

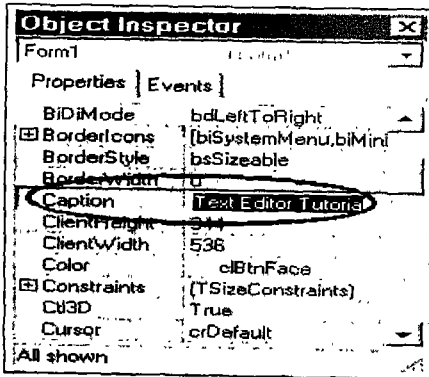


**تحديد قيم الخصائص**

عندما تفتح مشروع جديد تقوم لغة Delphi بعرض الفورمة الأساسية للمشروع والتي تسمى Form1 (وهو اسم افتراضى). وأنت ستقوم بإنشاء واجهة الاستخدام User Interface والأجزاء الأخرى من التطبيق الذى تتولى إعداده وذلك عن طريق وضع المكونات بهذه الفورمة.

بجوار الفورمة ستشاهد نافذة أداة فحص الكائن Object Inspector والتي من خلالها تستطيع أن تحدد قيم خصائص الفورمة والمكونات التى تضعها بالفورمة. وعندما تحدد قيم الخصائص تقوم لغة Delphi بإعداد الكود البرمجى المصدري الخاص بهذا التحديد بدلا من أن تقوم بهذه المهمة بنفسك. ونود هنا القول بأن القيم التى تحددها فى نافذة Object Inspector يطلق عليها قيم تحديدية لفترة التصميم Design-Time settings. وللقيام بذلك اتبع الخطوات التالية :

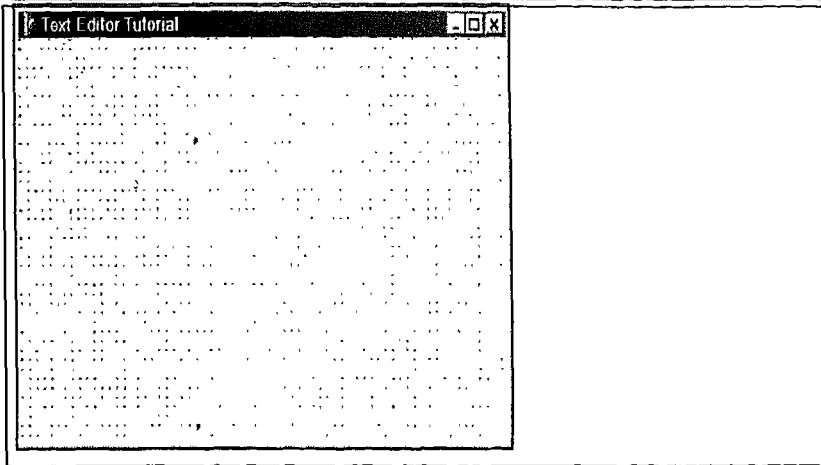
(١) فى نافذة Object Inspector ابحث عن الخاصية Caption للفورمة وعندها اكتب Text Editor Tutorial ليكون بديل للقيمة الافتراضية Form1 لهذه الخاصية كما هو موضح بالشكل التالى :



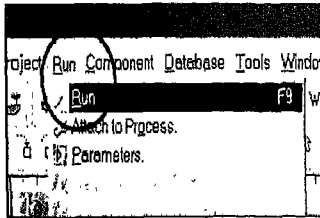
شكل توضيحي :

لاحظ أن القيمة التى يتم تخصيصها للخاصية Caption للفورمة تظهر فى شريط العنوان Title bar لهذه الفورمة كما هو موضح بالشكل التالى :



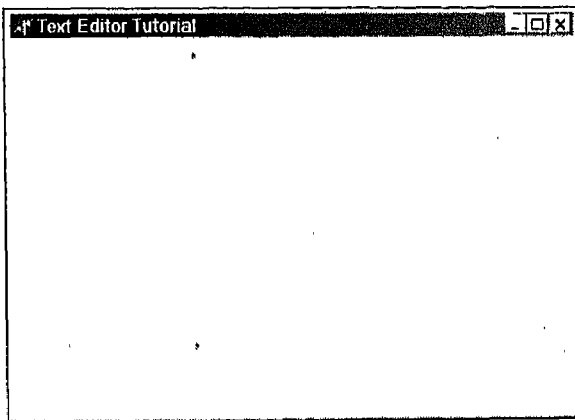


(٢) الآن قم بتشغيل الفورمة (حتى ولو كانت غير مشتملة على أى مكونات) عن طريق الضغط على المفتاح F9 بلوحة المفاتيح أو فتح القائمة Run واختيار الأمر Run منها كما هو موضح بالشكل التالى :



شكل توضيحي :

والآن تظهر الفورمة على الشاشة بالوضع الموضح فى الشكل التالى :



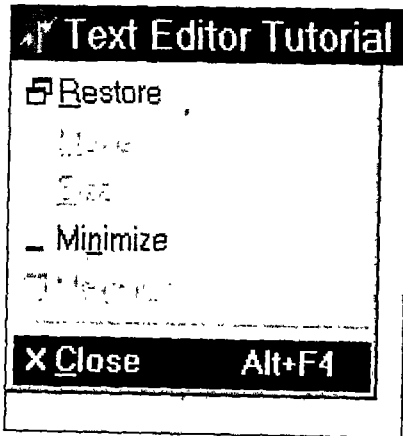
شكل توضيحي :



(٣) لكي تعود مرة أخرى إلى مشهد الفورمة Form1 في مرحلة التصميم قم بتنفيذ
أى من الآتى :

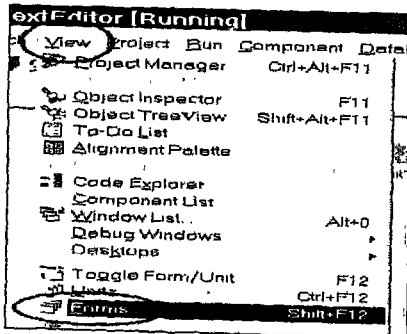
❶ انقر بالماوس على الزر (X) فى الجانب الأيمن لشريط عنوان فورمة التطبيق (مشهد التشغيل runtime view للفورمة والموضح فى الشكل السابق).

❷ انقر بالماوس على الأيقونة الموجودة على يسار شريط العنوان لتظهر قائمة تحكم ومنها اختر Close كما هو موضح بالشكل التالى :



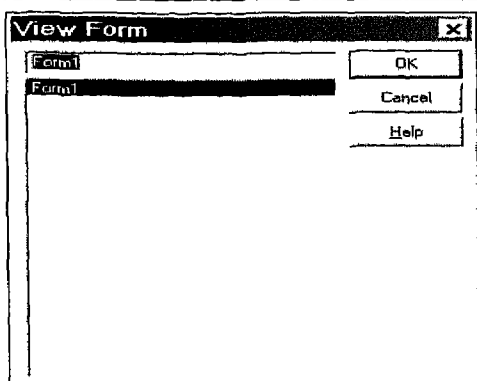
شكل توضيحي :

❸ افتح القائمة View ثم اختر منها Forms ثم اختر Forms كما هو موضح بالشكل
التالى (أو اضغط على المفاتيح Shift+F12 بلوحة المفاتيح):



شكل توضيحي :

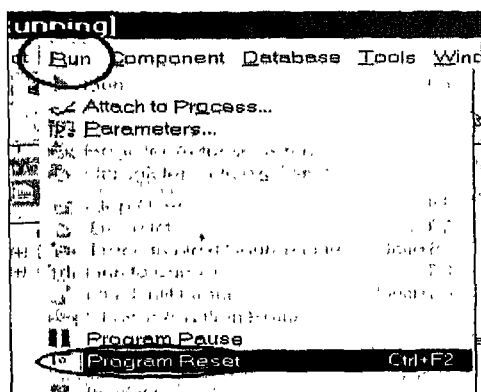
ليظهر على الشاشة صندوق الحوار View Form الموضح فى الشكل التالى :



شكل توضيحي :

بصندوق الحوار هذا اختر Form1 ثم انقر بالماوس على المفتاح Ok.

افتح القائمة Run واختر منها الأمر Program Reset كما هو موضح بالشكل التالي (أو اضغط على المفاتيح Ctrl+F2 بلوحة المفاتيح):



شكل توضيحي :

إضافة المكونات إلى الفورمة

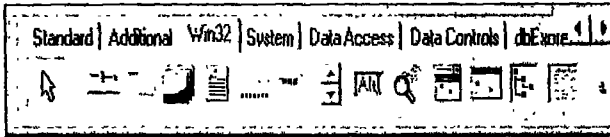
قبل أن تبدأ في إضافة المكونات إلى الفورمة ستحتاج للتفكير في أبسط طريقة لإنشاء واجهة الاستخدام UI (اختصار للمصطلح User Interface) للتطبيق الذي تتولى إعداده. وواجهة الاستخدام UI هي التي تسمح للمستخدم الذي يتعامل مع التطبيق الذي تعده بأن يتفاعل مع العناصر التي يتألف منها التطبيق وبالتالي يجب تصميم واجهة الاستخدام UI بحيث تكون سهلة الاستخدام.

تتضمن لغة Delphi العديد من المكونات التي تمثل الأجزاء التي يتألف منها أى تطبيق يتم إعداده بلغة Delphi. فعلى سبيل المثال هناك مكونات (منحدره من كائنات) بباليته المكونات التي تجعل من السهولة بمكان إضافة العديد من العناصر المرئية والغير مرئية للبرنامج مثل القوائم Menus وشرائط الأدوات toolbars و صناديق الحوار وغيرها الكثير والكثير...

تطبيق محرر النصوص الذى تحن بصدده هنا يتطلب وجود منطقة لتحرير النصوص وشريط حالة Status bar لعرض بعض المعلومات مثل اسم الملف الذى يتم تحريره الآن. كما يحتاج أيضا لمجموعة من القوائم Menus. كذلك من الممكن أن يشتمل على شريط أدوات يضم مجموعة من الأيقونات لسهولة الوصول للأوامر الموجودة بالقوائم. هذا ونود هنا القول بأن جمال تصميم واجهة الاستخدام UI باستخدام لغة Delphi يتمثل فى أنك تستطيع التعامل مع العديد من المكونات المختلفة ومشاهدة النتائج المترتبة على ذلك على الفور. وبهذه الطريقة تستطيع بكل سرعة إعداد واجهة الاستخدام UI لأى تطبيق بأقل قدر من المجهود.

لكى تبدأ فى تصميم محرر النصوص عليك أن تضيف كل من المكون RichEdit والمكون StatusBar للفورمة وذلك من خلال الخطوات التالية :

- (١) لكى تنشأ منطقة للنصوص عليك أولا أن تضيف المكون RichEdit. ولكى تعثر على هذا المكون انقر بالماوس على الصفحة Win32 بباليته المكونات لتظهر على السطح كما هو موضح بالشكل التالى :

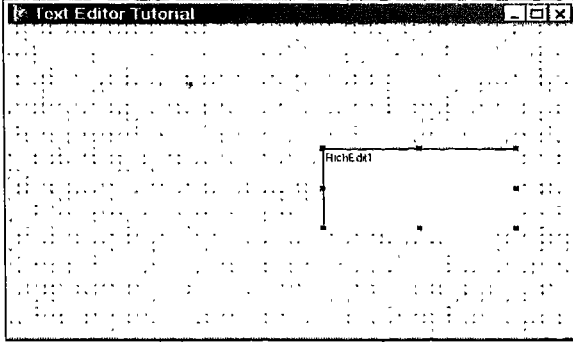


شكل توضيحي :

ثم ضع مؤشر الماوس على الأيقونة RichEdit. وعندما تعثر على المكون RichEdit فى هذه الحالة يمكن أن تقوم بأى من الآتى :

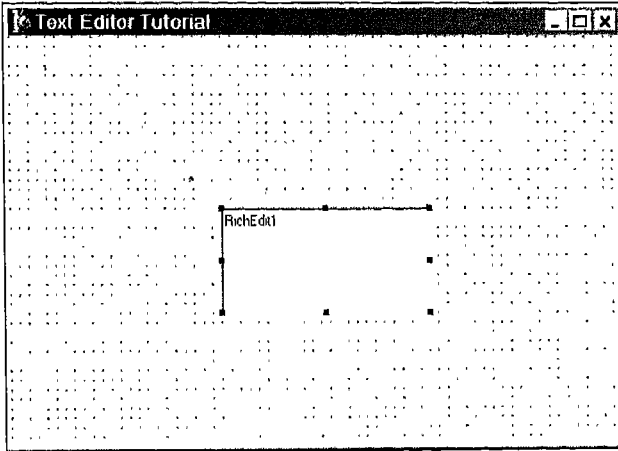


تنقر بالماوس على أيقونة المكون RichEdit بباليته المكونات ثم تنقر بالماوس على الموضع الذى ترغب فى وضع المكون به داخل الفورمة ليتم وضعه بهذا الموضع كما هو موضح بالشكل التالى :



شكل توضيحي :

أو تنقر بالماوس نقرا مزدوجا على أيقونة المكون بباليته المكونات ليتم وضعه فى مركز الفورمة كما هو موضح بالشكل التالى :

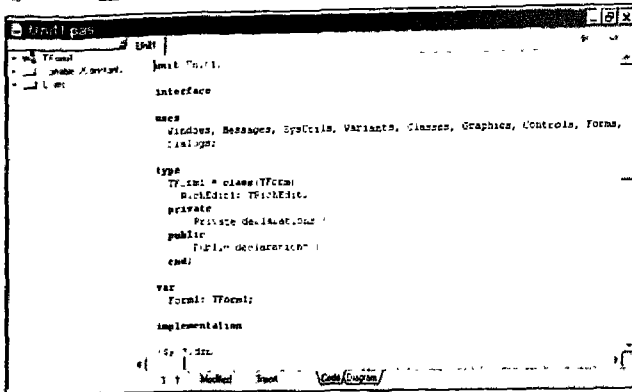


شكل توضيحي :

كل مكون من المكونات المتاحة لدى لغة Delphi يعتبر قطاع كما أن وضع مكون فى فورمة يؤدي إلى إنشاء حالة من هذا القطاع. هذا وبمجرد أن يصبح المكون فى الفورمة تقوم لغة Delphi بتكوين الكود البرمجي الضرورى لتوجيه حالة الكائن عند تشغيل التطبيق. والشكل التالى يوضح لنا الكود البرمجي الذى تم تكوينه للمكون RichEdit :

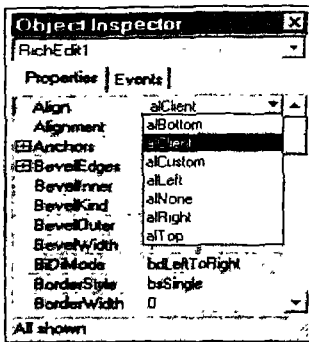


شكل توضيحي :



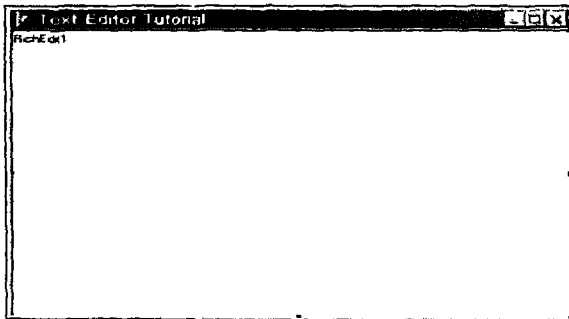
(٢) فى أثناء كون المكون RichEdit مختار (معلم عليه) بالفورمة قم بالبحث عن الخاصية Align بالنافذة Object Inspector ثم افتح القائمة المنسدلة لهذه الخاصية واختر منها alClient كما هو موضح بالشكل التالى :

شكل توضيحي :



تلاحظ الآن أن المكون RichEdit قد ملء الفورمة بأكملها كما هو موضح بالشكل التالى :


شكل توضيحي :

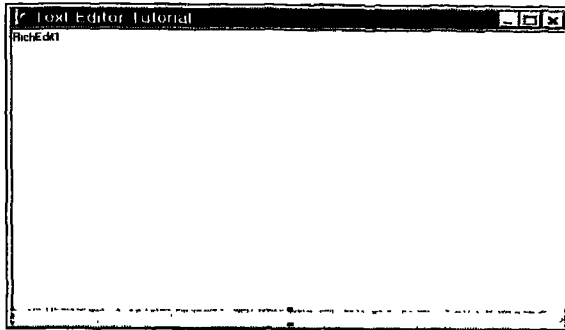


ومن ثم فقد أصبح لديك الآن منطقة كبيرة لتحرير النصوص.

باختيار القيمة alClient من القائمة المنسدلة للخاصية Align يصبح حجم أداة التحكم RichEdit قابل للتغيير ويتحدد على حسب حجم الفورمة الموجودة بها أداة التحكم RichEdit وذلك في حالة كون الفورمة تتمتع بإمكانية تغيير الحجم.



(٣) انقر بالماوس نقرا مزدوجا على أيقونة المكون StatusBar  بالصفحة Win32 بباليته المكونات. مما يؤدي إلى إضافة شريط حالة الجزء السفلي للفورمة كما هو موضح بالشكل التالي :



شكل توضيحي :

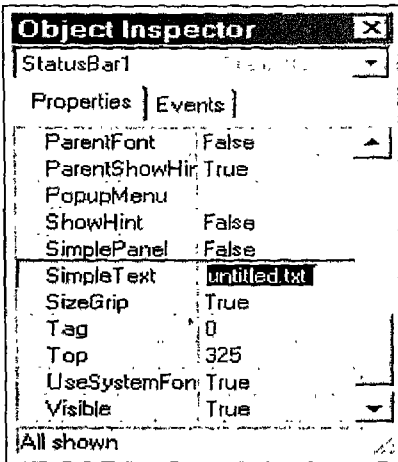
(٤) لكي تنشأ panel واحدة بشريط الأدوات لعرض اسم وموضع الملف الذي يتم تحريره من خلال محرر النصوص الذي نعهه الآن عليك إذن القيام بالآتي :

● تأكد من أن شريط الحالة مختار (معلم عليه) في الفورمة كما هو موضح بالشكل السابق.

● في النافذة Object Inspector ابحث عن الخاصية SimpleText وعندها اكتب untitled.txt كما هو موضح بالشكل التالي :



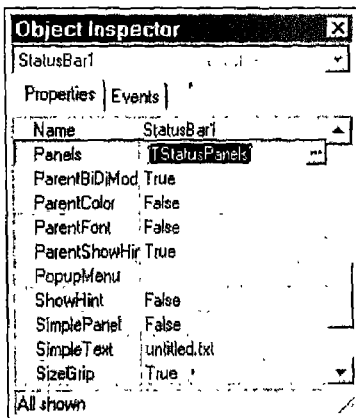
شكل توضيحي :



نتيجة ما سبق أنه عندما تستخدم محرر النصوص لو كان الملف الذي يتم تحريره لم يتم حفظه بعد فإن اسم الملف في هذه الحالة سيكون .untitled.txt



انقر بالماوس على الأيقونة (...) الموجودة بجوار القيمة TstatusPanel عند الخاصية Panels كما هو موضح بالشكل التالي :

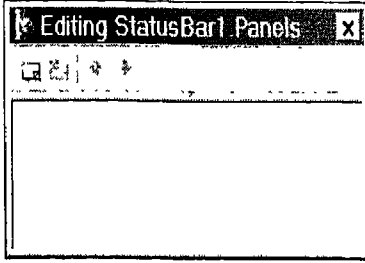



شكل توضيحي :

ليظهر على الشاشة صندوق الحوار Editing StatusBar1.Panels الموضح في الشكل التالي :

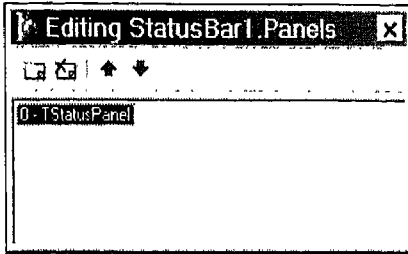


شكل توضيحي :



بشريط الأدوات الخاص بصندوق الحوار هذا انقر بالماوس على الأيقونة New Action  ليتم إضافة Panel بشريط الحالة كما هو موضح بالشكل التالي :

شكل توضيحي :



تستطيع أيضا الوصول لصندوق الحوار Editing StatusBar1.Panels عن طريق النقر بالماوس نقرا مزدوجا على شريط الحالة الموجود بالفورمة.



(٥) انقر بالماوس على الزر (X) لغلاق صندوق الحوار Editing StatusBar1.Panels.

لقد إنتهيت الآن من إعداد وتهيئة واجهة الاستخدام UI لمحرر النصوص الذى تتولى إعداده.

إضافة الدعم الخاص بالقوائم وشرائط الأدوات

لجعل التطبيق لديه القدرة على عمل أى شئ فإنه يحتاج لقائمة ومجموعة من الأوامر ومن الأفضل إضافة شريط أدوات لهذا التطبيق. وبالرغم من أنك تستطيع كتابة كود برمجى لكل أمر من الأوامر على حدة إلا إن لغة Delphi تعمل على توفير ما يعرف بمدير الفعل Action Manager للمساعدة فى جعل الكود البرمجى كله مجتمعا فى موضع واحد وهو ما يعرف بمركزية الكود البرمجى centralize the code بالإضافة لإعداد قائمة صورة



image list لمركزية الصور لإضافتها للأوامر الموجودة بالقوائم وشريط الأدوات بالتطبيق الذي تتولى إعداده.

من الأنسب أن يتم تسمية الأفعال المتصلة بالأوامر الموجودة بالقائمة مستعينا بكل من اسم القائمة واسم الأمر. فعلى سبيل المثال الفعل المسمى FileExit يشير إلى الأمر Exit الموجود بالقائمة File وهكذا...

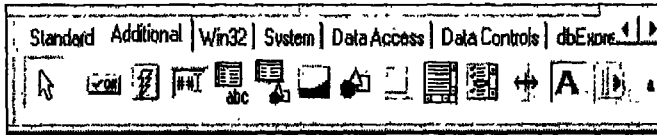
الجدول التالي يقدم أنواع الأفعال التي يحتاج إليها تطبيق محرر النصوص الذي نتولى إعداده :

القائمة	الأمر	أيقونة بشريط الأدوات؟	الوصف
File	New	Yes	إنشاء ملف جديد.
File	Open	Yes	فتح ملف سبق إنشاؤه لتحرير محتوياته.
File	Save	Yes	حفظ الملف الذي يتم التعامل معه حالياً.
File	Save As	No	حفظ ملف باستخدام اسم جديد (كذلك السماح لك بملف جديد باستخدام اسم يتم اقتراحه).
File	Exit	Yes	إنهاء التعامل مع محرر النصوص.
Edit	Cut	Yes	استقطاع النص المختار ووضعه في لوحة الاقتباس clipboard الخاصة ببيئة الويندوز.
Edit	Copy	Yes	عمل نسخة من النص المختار ووضعها في لوحة الاقتباس clipboard الخاصة ببيئة الويندوز.

وضع النص المخزن في لوحة الاقتباس clipboard الخاصة ببيئة الويندوز بالموضع الموجود به مؤشر الكتابة بمنطقة تحرير النص.	Yes	Paste	Edit
عرض شاشة محتويات نظام المساعدة والتي من خلالها تستطيع الوصول لمواضيع المساعدة.	No	Contents	Help
عرض شاشة دليل المساعدة.	No	Index	Help
عرض صندوق حوار يشتمل على معلومات حول التطبيق.	No	About	Help

لمركزية كل من الكود البرمجي والصور في مدير أى فعل فإنك تحتاج لإضافة محرر مدير الفعل Action Manager إلى المشروع الذى تعده وذلك من خلال الخطوات التالية :

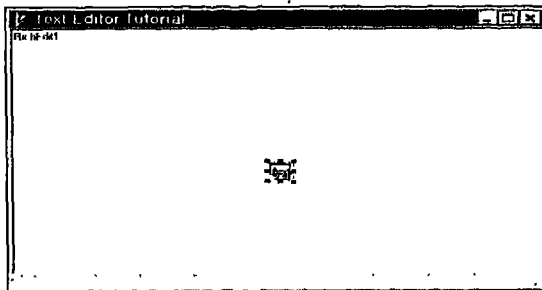
(١) انقر بالماوس على الصفحة Additional بباليطة المكونات لتظهر على السطح كما هو موضح بالشكل التالى :



شكل توضيحي :

(٢) بهذه الصفحة انقر بالماوس نقرا مزدوجا على أيقونة المكون ActionManager

لإسقاطه فى القورمة كما هو موضح بالشكل التالى :

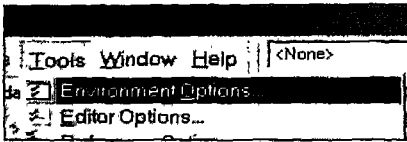


شكل توضيحي :

بسبب أن هذا المكون من المكونات التي تكون مختلفة في مرحلة التشغيل Run-Time لذلك تستطيع وضعه في أي مكان بالفورمة.

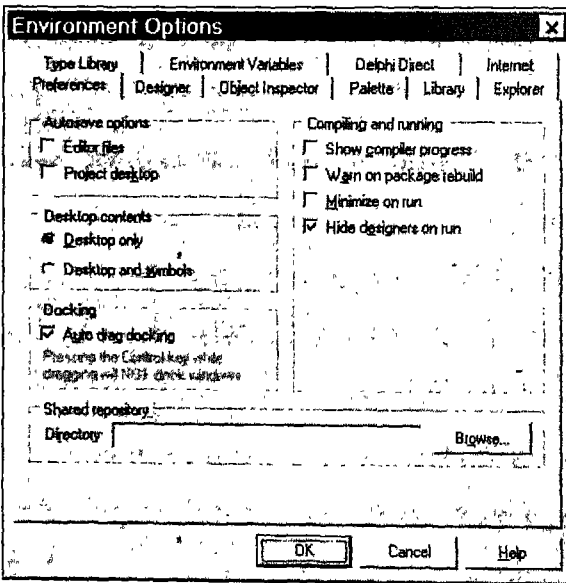


(٣) لعرض الخيارات الخاصة بالمكونات التي تكون مختلفة في مرحلة التشغيل Run-Time والتي تقوم بوضعها في الفورمة افتح القائمة Tools واختار منها الأمر Environment كما هو موضح بالشكل التالي :



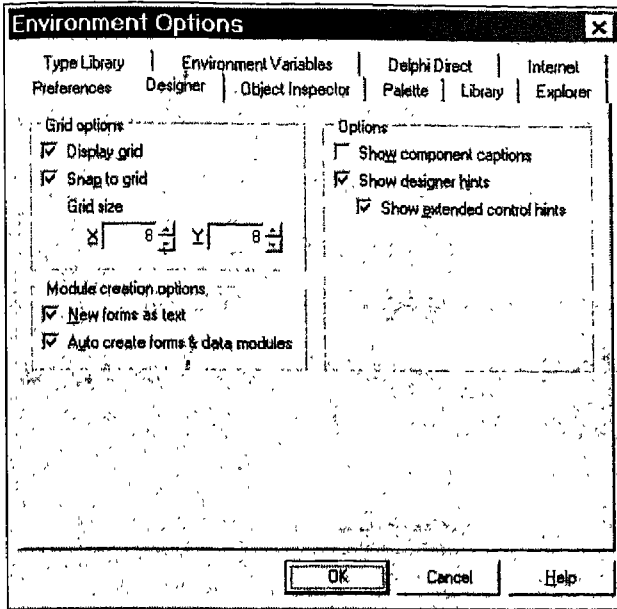
شكل توضيحي :

(٤) ليظهر على الشاشة صندوق الحوار Environment Options الموضح في الشكل التالي :



شكل توضيحي :

(٥) انقر بالماوس على التبويب Designer ليظهر على السطح كما هو موضح بالشكل التالي :



شكل توضيحي :

(٦) بهذا التيويب علم بالماوس على الاختيار Show component captions ثم انقر بالماوس على المفتاح Ok.

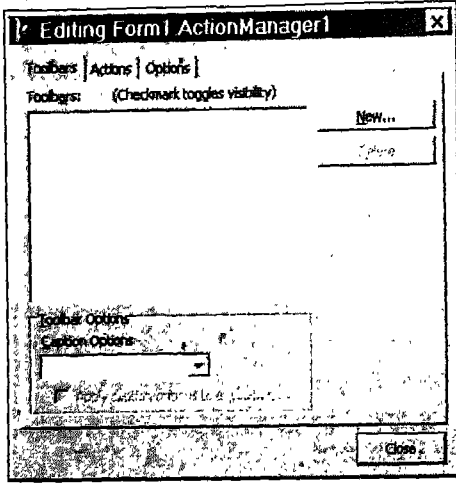
إضافة الأفعال إلى مدير الأفعال Action Manager

فى البداية ستقوم بإضافة الأفعال إلى مدير الأفعال كما ستحدد خصائص كل فعل من هذه الأفعال. ومن المناسب فى هذه المرحلة أن تكون أسماء الأفعال التى تتصل بالأوامر الموجودة بالقوائم مؤلفة من اسم القائمة ومن اسم الأمر معا. فعلى سبيل المثال الفعل المسمى FileExit يشير إلى الأمر Exit الموجود بالقائمة File.

ستقوم الآن بإضافة نوعين من الأفعال الأولى سيتم تحديد خصائصها بنفسك والثانية تعرف بأنها أفعال قياسية يتم تحديد خصائصها تلقائيا. وللقيام بذلك اتبع الخطوات التالية :

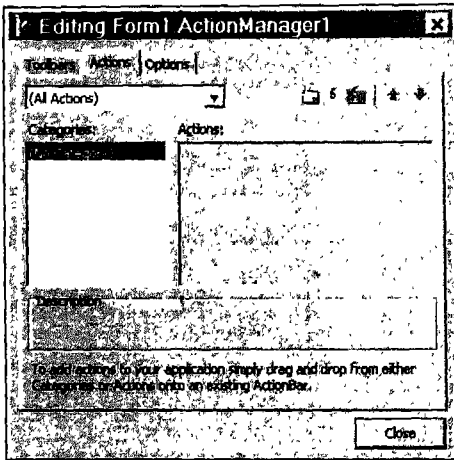
(١) انقر بالماوس نقر مزدوجا على المكون ActionManager الموجود بالفورمة ليظهر على الشاشة صندوق الحوار الموضح فى الشكل التالى :

شكل توضيحي :



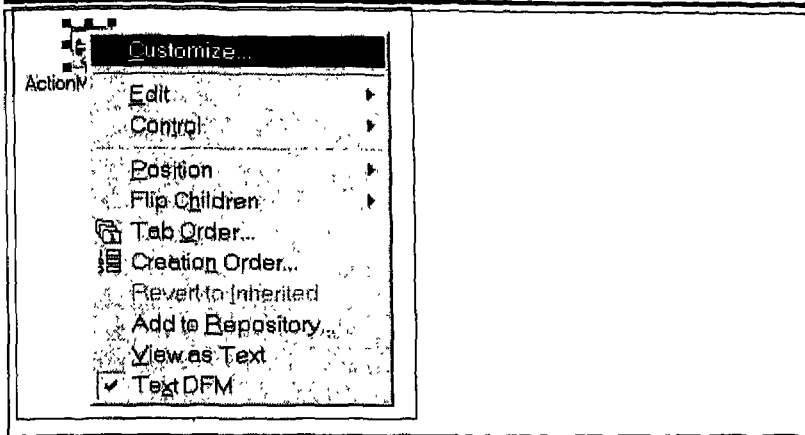
(٢) تأكد من أن التبويب Actions ظاهر على السطح وإلا فانقر عليه بالماوس لجعله ظاهر على السطح كما هو موضح بالشكل التالي :


شكل توضيحي :

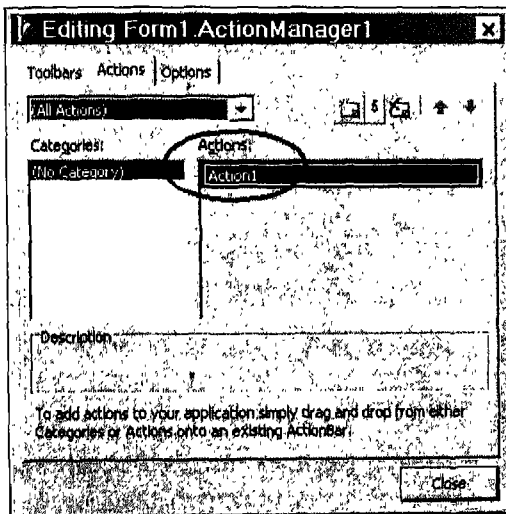


تستطيع أيضا النقر بالزر الأيمن للماوس على أيقونة المكون ActionManager بالفورمة ثم اختيار الأمر Customize من القائمة التي تظهر بجوار مؤشر الماوس كما هو موضح بالشكل التالي :





(٣) في أثناء كون No Category مختار في قائمة العرض Categories انقر بالماوس على الأيقونة New Action  ليظهر Action1 في قائمة العرض Actions كما هو موضح بالشكل التالي :

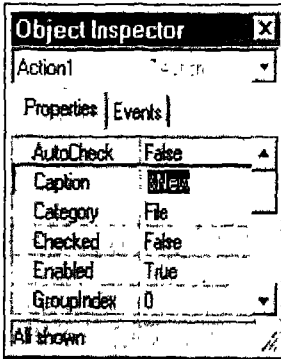


شكل توضيحي :

في نافذة Object Inspector قم بتحديد الخصائص التالية للفعل Action1 :

عند الخاصية Caption اكتب &New كما هو موضح بالشكل التالي :

شكل توضيحي :

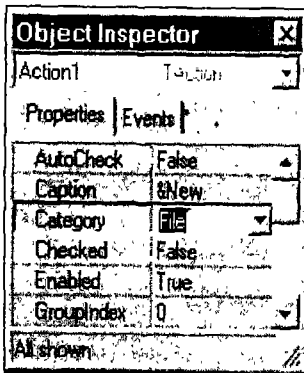


لاحظ أن كتابة العلامة & قبل حرف معين يؤدي إلى جعل هذا الحرف تحته خط في مرحلة التشغيل وفي هذه الحالة يعرف هذا الحرف بأنه حرف الاستدعاء.



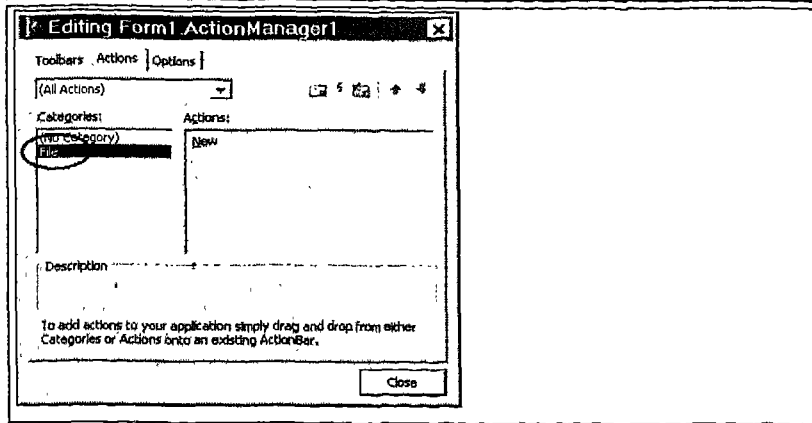
عند الخاصية Category اكتب File كما هو موضح بالشكل التالي :

شكل توضيحي :

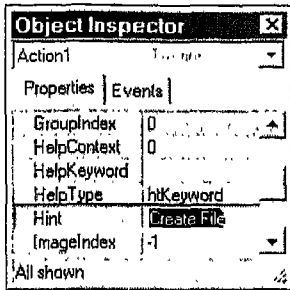


هذا التحديد يعمل على تنظيم الأوامر الخاصة بالقائمة File ووضعها في موضع واحد. كما إنه يؤدي إلى ظهور File في قائمة العرض Categories بصندوق الحوار كما هو موضح بالشكل التالي :





عند الخاصية Hint اكتب Create file كما هو موضح بالشكل التالي :

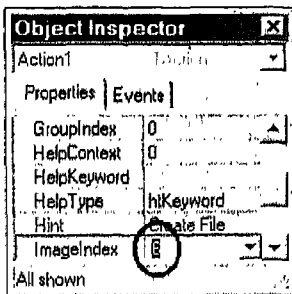


شكل توضيحي :

قيمة هذه الخاصية ستظهر في أداة التعليق ToolTip التي تظهر عند وضع الماوس على الأمر New بالقائمة File لفترة بسيطة في مرحلة التشغيل Run-Time.



عند الخاصية ImageIndex اكتب القيمة ٦ كما هو موضح بالشكل التالي :

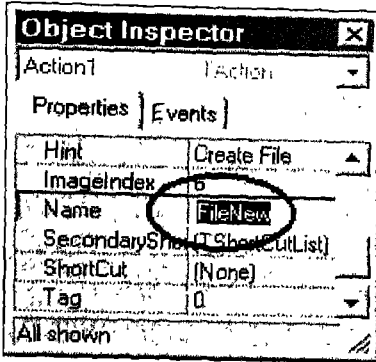


شكل توضيحي :

هذا التحديد سيربط الصورة رقم (٦) في قائمة الصور ImageList (التي ستقوم بإعدادها) بهذا الفعل.



عند الخاصية Name اكتب FileNew كما هو موضح بالشكل التالي :



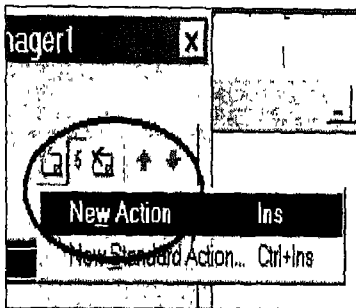
شكل توضيحي :

هذا الاسم سيكون اسم الفعل المرتبط بالأمر New الموجود بالقائمة File.



الآن اضغط على المفتاح Enter بلوحة المفاتيح لحفظ القيم التي تم تخصيصها لهذه الخصائص.

(٤) بصندوق الحوار Action Manager انقر بالماوس على السهم المجاور للأيقونة New Action ثم اختر New Action كما هو موضح بالشكل التالي :



شكل توضيحي :

(٥) انقر بالماوس على Action1 في قائمة العرض Actions ثم في نافذة Object Inspector قم بتحديد قيم الخصائص التالية :

- عند الخاصية Caption اكتب &Save.
- عند الخاصية Category افتح القائمة المنسدلة واختر منها File..
- عند الخاصية Hint اكتب Save File.
- عند الخاصية ImageIndex اكتب القيمة ٨.
- عند الخاصية Name اكتب FileSave (للأمر Save بالقائمة File).
- (٦) كرر الخطوة رقم (٤).
- (٧) كرر الخطوة رقم (٥) ولكن مع الخصائص التالية :
- عند الخاصية Caption اكتب &Index.
- عند الخاصية Category اكتب Help.
- عند الخاصية Name اكتب HelpIndex (ليكون اسم الفعل الذي يشير للأمر Index الموجود بالقائمة Help).
- (٨) كرر الخطوة رقم (٤).
- (٩) كرر الخطوة رقم (٥) ولكن مع الخصائص التالية :
- عند الخاصية Caption اكتب &About.
- عند الخاصية Category افتح القائمة المنسدلة واختر منها Help.
- عند الخاصية Name اكتب HelpAbout (ليكون اسم الفعل الذي يشير للأمر About الموجود بالقائمة Help).
- (١٠) لا تقم بإغلاق صندوق الحوار Action Manager.

إضافة الأفعال القياسية لقائمة الأفعال

فيما يلي ستقوم بإضافة الأفعال القياسية (الفتح Open والحفظ باسم Save As وإنهاء البرنامج Exit والقص Cut والنسخ Copy واللصق Paste ومحتويات نظام المساعدة Help Contents) إلى مدير الأفعال Action Manager. وللقيام بذلك اتبع الخطوات التالية :

(١) ينبغي أن يكون مدير الأفعال Action Manager لا يزال مفتوحا ومعروضا على الشاشة. وإذا لم يكن كذلك انقر بالماوس نقرًا مزدوجًا على المكون ActionManager بالفورمة.

(٢) انقر بالماوس على السهم المجاور للأيقونة New Action ثم اختر New Standard Action كما هو موضح بالشكل التالي :



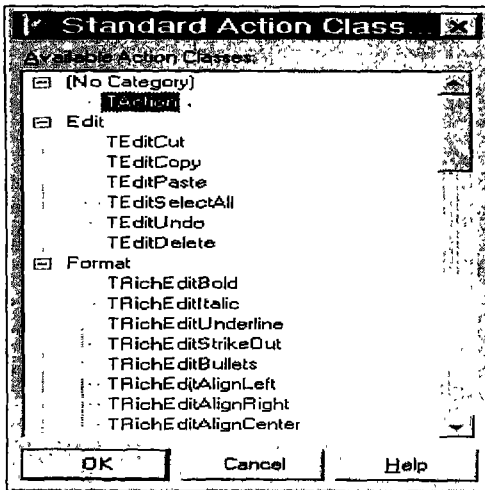
شكل توضيحي :

يمكن أيضا أن تستخدم الاختيار New Standard Action بالضغط على المفاتيح Ctrl+Ins بلوحة المفاتيح.



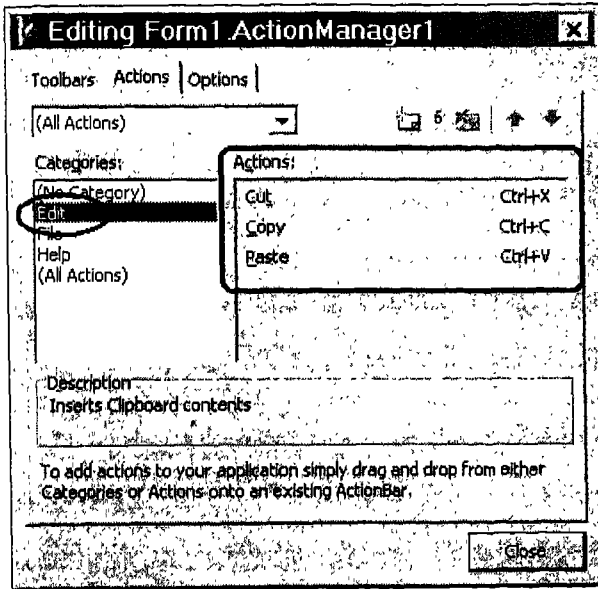
ليظهر على الشاشة صندوق الحوار Standard Actions Classes الموضح في

الشكل التالي :



شكل توضيحي :

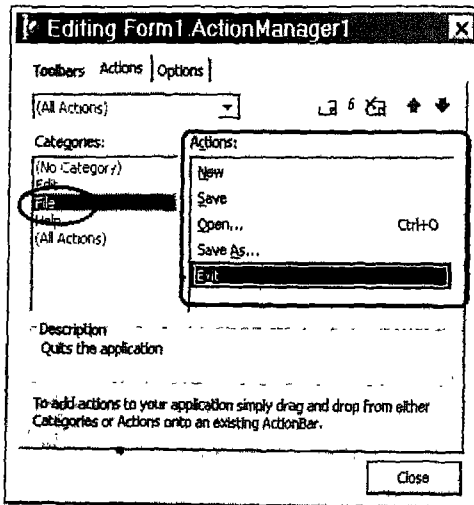
(٣) في صندوق الحوار Standard Actions Classes تحرك داخل قائمة العرض حتى تصل إلى القسم Edit ثم اختر كل من TEditCut و TEditCopy و TEditPaste (في اثناء ذلك استخدم المفتاح Ctrl بلوحة المفاتيح عند النقر على هذه الأفعال بالماوس ليتم إختيارهم معا) ثم انقر بالماوس على المفتاح Ok ليتم إضافة هذه الأفعال إلى القسم Edit الجديد بقائمة العرض Categories بصندوق الحوار Action Manager كما هو موضح بالشكل التالي :



شكل توضيحي :

(٤) كرر الخطوة رقم (٢) مرة أخرى لفتح صندوق الحوار Standard Actions Classes.

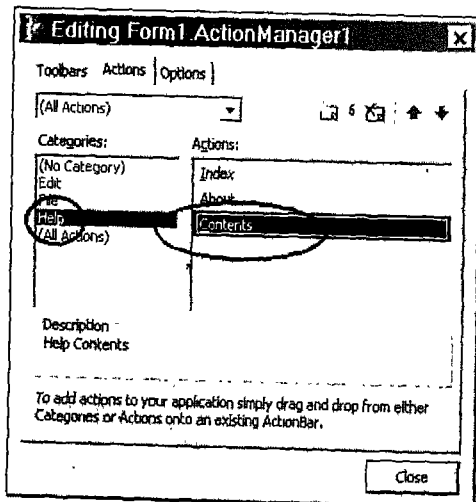
(٥) بصندوق الحوار هذا ابحث عن القسم File وعندما تصل إليه اختر الأفعال TFileOpen و TFileSaveAs و TFileExit ثم انقر بالماوس على المفتاح Ok لتجد أنه تم إضافة هذه الأفعال إلى القسم File في صندوق الحوار Action Manager كما هو موضح بالشكل التالي :



شكل توضيحي :

(٦) مرة أخرى كرر الخطوة رقم (٢) مرة أخرى لفتح صندوق الحوار Standard Actions Classes.

(٧) بصندوق الحوار هذا ابحث عن القسم Help وعندما تصل إليه اختر الفعل THelpContents ثم انقر بالماوس على المفتاح Ok لتجد أنه تم إضافة هذه الأفعال إلى القسم Help في صندوق الحوار Action Manager كما هو موضح بالشكل التالي :



شكل توضيحي :

إضافة الأمر Contents المفصل إلى القائمة Help سيؤدي إلى عرض ملف مساعدة يشتمل على التبويب Contents. أما الأمر Contents القياسي بالقائمة Help فيعمل على استحضار آخر تبويب تم التعامل معه آخر مرة سواء كان التبويب Contents أو التبويب Index أو التبويب Find.

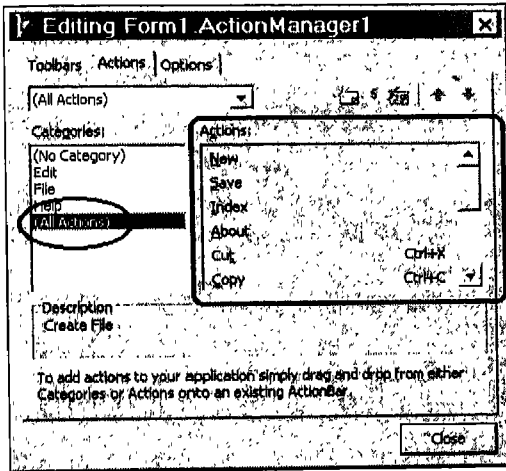


حتى الآن إنتهيت من إضافة كافة الأفعال القياسية التي يحتاج إليها تطبيق محرر النصوص الذي نتولى إعداده.

الأفعال القياسية لديها الخصائص الخاصة بها والتي يتم تحديد قيمتها تلقائياً بما فيها الخاصية ImageIndex.



(٨) انقر بالماوس على (All Actions) بالقسم Categories لتظهر في القائمة Actions كافة الأفعال التي أضفتها سواء كانت قياسية أو غير قياسية كما هو موضح بالشكل التالي :



شكل توضيحي :

(٩) انقر بالماوس على المفتاح Close لغلغ صندوق الحوار Action Manager.

(١٠) افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التي أجريتها بالمشروع.



إضافة الصور لقائمة الصور Image List

فى هذا الجزء من الفصل ستقوم بإضافة صور إلى مدير الأفعال من أجل استخدامها بكل من القوائم وشريط الأدوات.

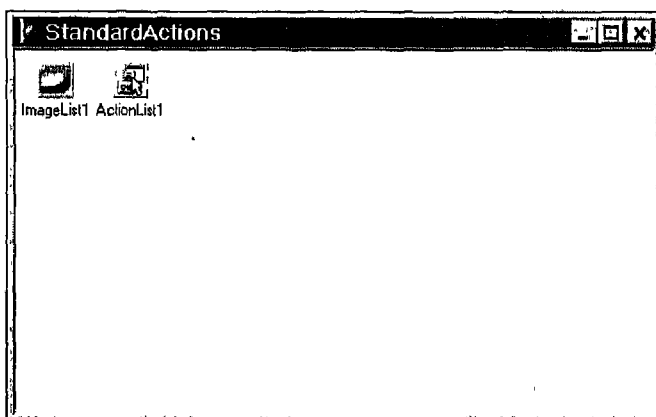
الأفعال القياسية يكون ملحق بها صوراً سابقة التخصيص وهذه الصور مستمدة من قائمة صور أساسية Built-In تأتي مع لغة Delphi. فعلى سبيل المثال دليل الصورة Image Index للفعل القياسى Cut بالقائمة Edit عبارة عن 0.

كافة الصور التى ستستخدمها لأوامر محرر النصوص الذى تتولى إعداده موجودة بقائمة الصور الأساسية السالفة الذكر.



لكى تضيف قائمة الصور اتبع الخطوات التالية :

- (١) لو أنك قمت بتركيب لغة Delphi فى الموضع الذى يفترضه لك صندوق حوار التركيب والتهيئة فى أثناء عملية التهيئة فى هذه الحالة قم بفتح الملف ActnRes.pas الموجود بالمسار C:\Program files\Borland\Delphi6\Source\VCL
ليتم فتح نافذة الأفعال القياسية StandardActions الموضح فى الشكل التالى :

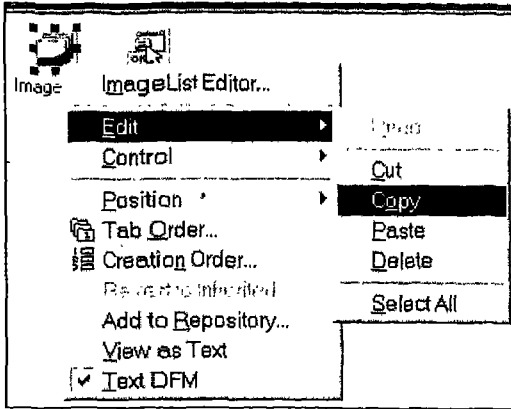


شكل توضيحي :

- (٢) اختر المكون ImageList1 ثم خذ منه نسخة بالنقر بالزر الأيمن على أيقونة هذا المكون ثم اختر Edit ثم اختر Copy كما هو موضح بالشكل التالى :

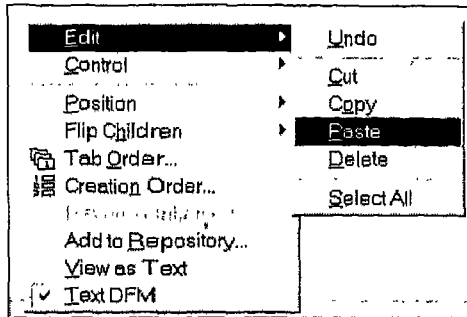


شكل توضيحي :



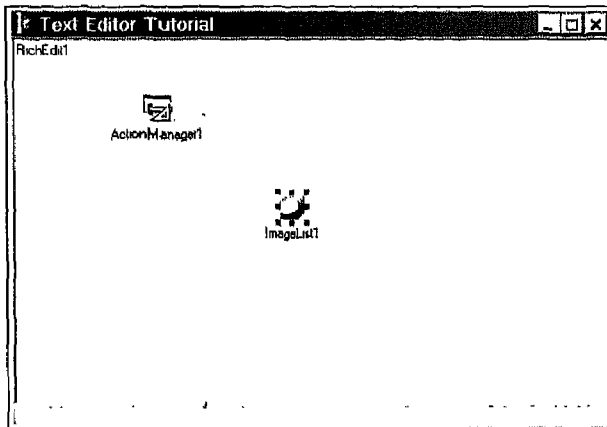
وفي الفورمة الخاصة بالمشروع الذى تعدده انقر بالزر الأيمن للماوس بأى موضع بالفورمة ثم اختر Edit ثم اختر paste كما هو موضح بالشكل التالى :

شكل توضيحي :



ليتم وضع المكون ImageList1 بالفورمة كما هو موضح بالشكل التالى :

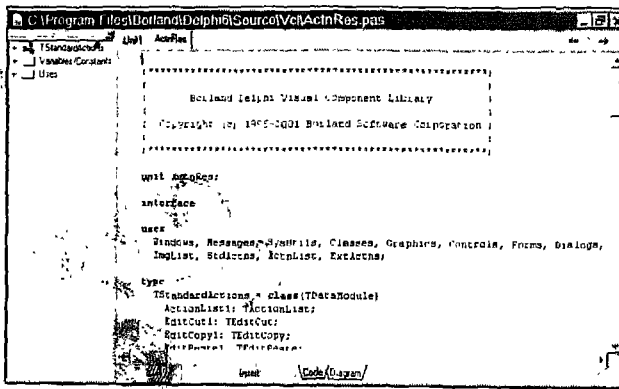
شكل توضيحي :



المكون **ImageList1** يعتبر من طائفة المكونات التي تكون مختلفة في مرحلة التشغيل **Run-Time** ومن ثم فلا يهم الموضع الذي تختاره لكي تلتصق فيه نسخة هذا المكون.



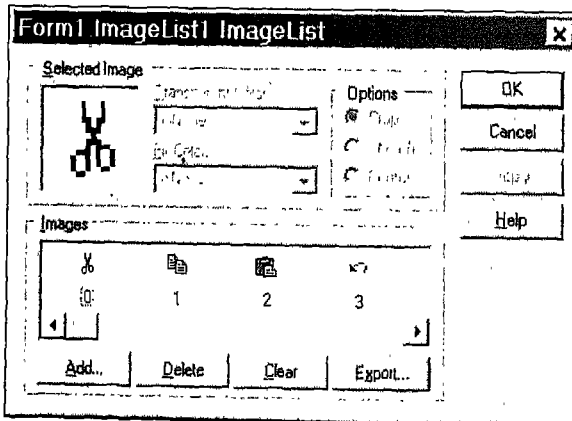
ستجد الآن أنه تم إضافة الوحدة **ActnRes.pas** إلى محرر الكود البرمجي كما هو موضح بالشكل التالي :



شكل توضيحي :

(٣) قم الآن بغلق النافذة **StandardActions**.

(٤) انقر بالماوس نقرا مزدوجا على الأيقونة **ImageList1** الموجودة بالفورمة ليظهر على الشاشة صندوق الحوار **ImageList** (الموضح في الشكل التالي) والذي يضم كافة الصور المتاحة والتي يمكن أن تستخدمها :

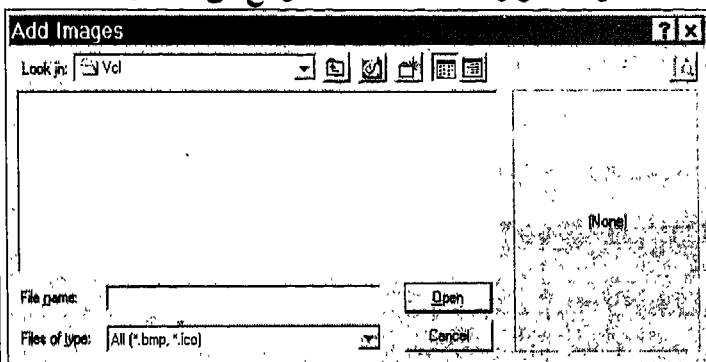


شكل توضيحي :

فيما يلي الجدول التالي يقدم لنا أرقام الصور المستخدمة مع كل أمر من الأوامر التي أضفتها لمحرر النصوص الذي تتولى إعداده :

القيمة الخاصة بـ ImageIndex	الأمر	القائمة
0	Cut	Edit
1	Copy	Edit
2	Paste	Edit
6	New	File
7	Open	File
8	Save	File
30	Save As	File
43	Exit	File
40	Contents	Help

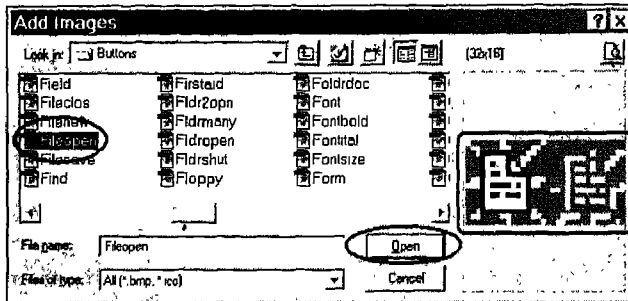
تستطيع أن تضيف صور من أي مجموعة مختلفة تماما. ففي صندوق الحوار Form1.ImageList انقر بالماوس على المفتاح Add ليظهر على الشاشة صندوق الحوار Add Images الموضح في الشكل التالي :



من خلال صندوق الحوار هذا ابحث عن الفهرس المسمى Buttons الموجود بنفس المسار الذي تم تركيب لغة Delphi به علما بأن الموضع الافتراضي لهذا الفهرس عبارة عن الآتي :

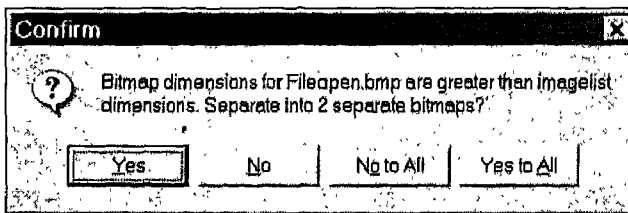
C:\Program Files\Common Files\Borland Shared\Images\Buttons.

فعلى سبيل المثال لو أردت أن تخصص صورة من الصور الموجودة بالمجلد Buttons للأمر Open الموجود بالقائمة File (بالتطبيق الذى تعده) فى هذه الحالة انقر بالماوس على الملف المسمى fileopen ثم على المفتاح Open كما هو موضح بالشكل التالى :



شكل توضيحي :

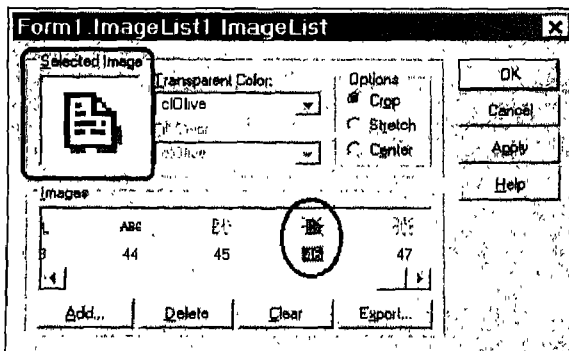
يظهر على الشاشة صندوق الحوار Confirm الموضح فى الشكل التالى :



شكل توضيحي :

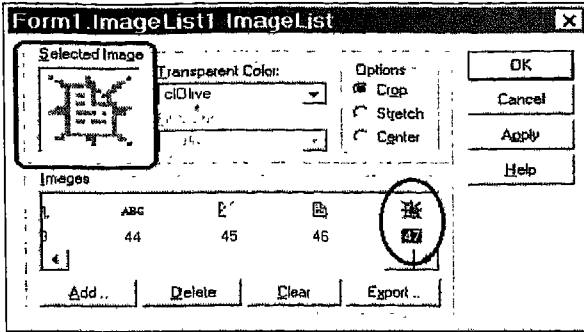
ليسألك عما إذا كنت ترغب فى تقسيم هذه الصورة إلى جزئين وفى حالتنا هذه انقر بالماوس على المفتاح Yes.

كل صورة من الصور لها وضع وهى نشطة Active ووضع آخر وهى غير نشطة. فعلى سبيل المثال الشكل التالى يوضح لنا وضع الصورة Fileopen وهى نشطة :



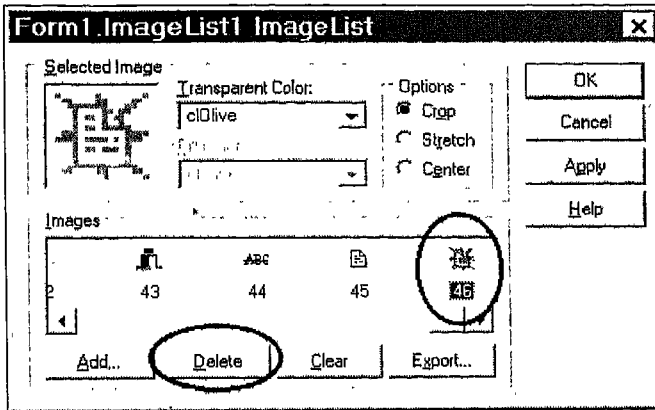
شكل توضيحي :

أما الشكل التالي فيوضح لنا وضع نفس الصورة وهي غير نشطة Grayed out :



شكل توضيحي :

والآن امسح وضع الصورة في حالة كونها غير نشطة Grayed Out (آخر صورة) والذي بالنقر عليها بالماوس ثم على المفتاح Delete كما هو موضح بالشكل التالي :



شكل توضيحي :

وبعد ذلك تأكد من أن دليل الصورة في النافذة Object Inspector يتطابق مع الرقم الجديد المخصص لهذه الصورة في القائمة التي تضم مجموعة الصور.

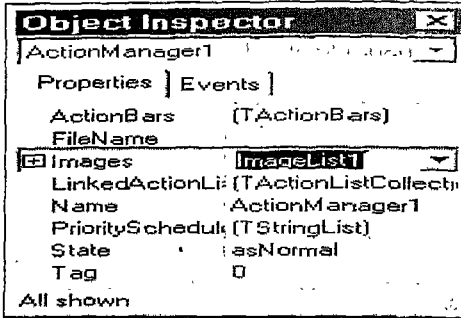
(٥) انقر بالماوس على المفتاح Ok لغلق صندوق الحوار ImageList.

(٦) اختر المكون ActionManager الموجود بالفورمة ثم عند الخاصية Images

الخاصة به افتح القائمة المنسدلة واختر منها ImageList1 كما هو موضح بالشكل التالي :



شكل توضيحي :



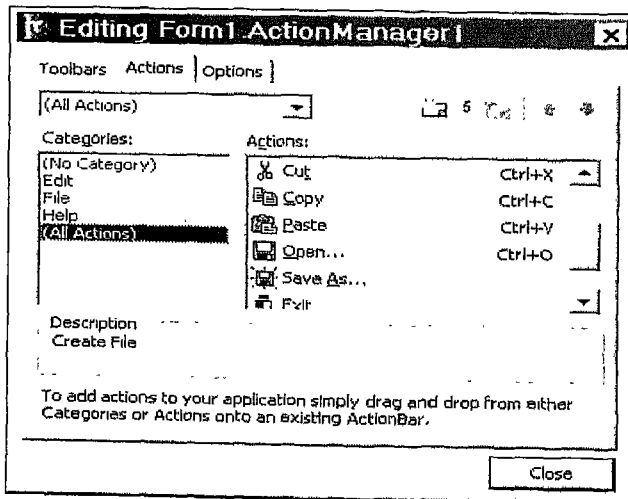
بسبب أنك قمت بالفعل بتحديد قيمة الخاصية ImageIndex لكافة الأفعال التي أضفتها لهذا التطبيق لذلك فإن كل صورة يتم إضافتها تلقائياً إلى الفعل الصحيح المناظر لها. ومن ثم فقد تم إلحاق ٨ صور للـ أفعال الموجودة بالتطبيق.



(٧) لكي تشاهد الصور الملحقة أو المرتبطة في مدير الأفعال قم بفتح المكون Action Manager الموجود بالفورمة وتأكد من أن التبويب Actions هو الذي يظهر على السطح ثم انقر بالماوس على القسم All Actions بقائمة العرض Categories لتشاهد صورة صغيرة بجوار كل فعل كما هو موضح بالشكل

التالي :

شكل توضيحي :





(٨) والآن افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التي أجريتها بالمشروع.


يمكن أيضا أن تضغط على مجموعة المفاتيح Shift+Ctrl+S بلوحة المفاتيح معا في نفس الوقت للقيام بنفس المهمة.

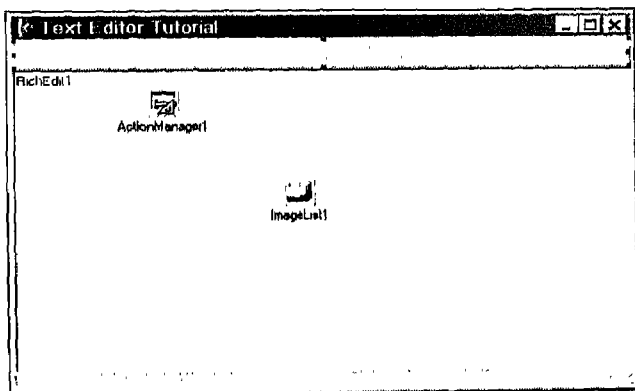


(٩) اجعل صندوق الحوار Action Manager مفتوحا.
الآن أنت على أهبة الاستعداد لكي تضيف القوائم وشريط الأدوات للمشروع.

إضافة قائمة Menu للمشروع

في خلال هذا الجزء والجزء القادم بهذا الفصل ستقوم بإضافة قائمة قابلة للتفصيل وشريط أدوات إلى المشروع الذي تعده وكلاهما يطلق عليه Action Bands.
شريط القائمة الأساسية يكون متضمنا ثلاثة قوائم وهي القائمة File والقائمة Edit والقائمة Help وكل منها تشتمل على الأوامر الخاصة بها. هذا ومع صندوق الحوار Action Manager تستطيع أن تسحب قسم كل قائمة (من قائمة العرض Categories) وما تحتويه من أوامر وإسقاطه في شريط القائمة (كل ذلك في خطوة واحدة فقط). وللقيام بذلك اتبع الخطوات التالية :

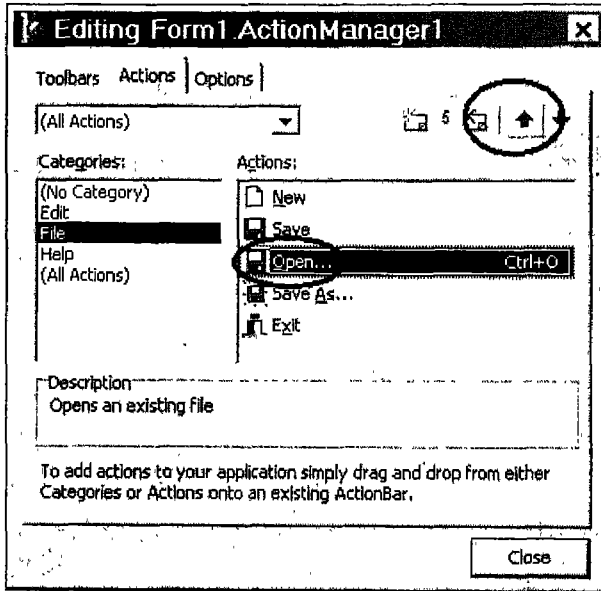
(١) من الصفحة Additional بباليئة المكونات انقر بالماوس نقرا مزدوجا على المكون ActionManinMenuBar  ليتم إضافتها إلى الفورمة كما هو موضح بالشكل التالي :



شكل توضيحي :

(٢) افتح صندوق الحوار Action Manager إذا لم يكن مفتوح بالفعل ثم انقر بالماوس على القسم File فى قائمة العرض Categories. وستلاحظ أن مجموعة الأوامر الخاصة بالقسم غير مرتبة بالترتيب الذى تطلبه بالضبط. ولكن على كل حال تستطيع بكل سهولة أن تقوم بتغيير هذا الترتيب وذلك عن طريق استخدام كل من الأيقونة Move Up (أو الضغط على المفاتيح Ctrl+Pgup بلوحة المفاتيح) والأيقونة Move Down (أو الضغط على المفاتيح Ctrl+PgDown بلوحة المفاتيح).

اختر الفعل Open ثم انقر بالماوس على الأيقونة Move Up كما هو موضح بالشكل التالى :

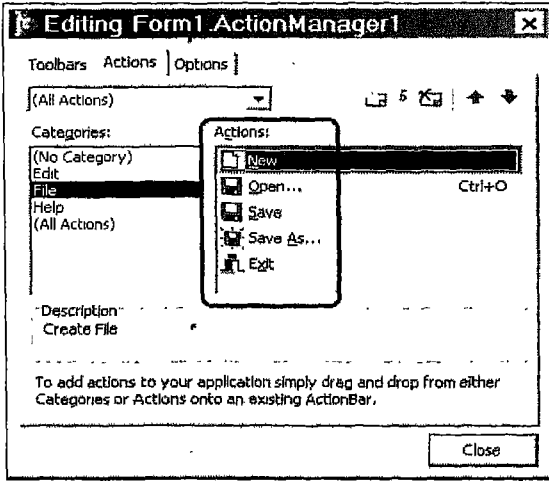


شكل توضيحي :

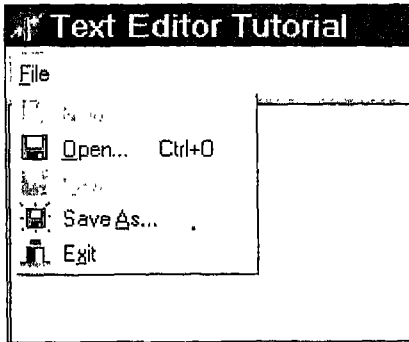
ومن ثم اجعل الأفعال الموجودة فى القسم File مرتبة كالآتى : New ثم Open ثم Save ثم Save As ثم Exit كما هو موضح بالشكل التالى :



شكل توضيحي :



(٣) اسحب القسم File من قائمة العرض Categories بصندوق الحوار Action Manager ثم اسقطه في شريط القائمة لتجد أنه قد ظهرت القائمة File بالأوامر الخاص به بشريط القائمة كما هو موضح بالشكل التالي :

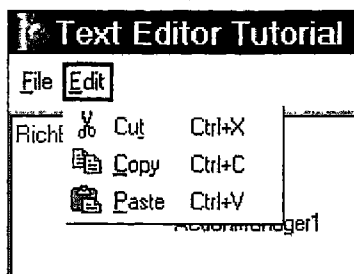


شكل توضيحي :

تستطيع أيضا أن تعيد ترتيب الأوامر بالقوائم وذلك بعد أن تنتهي من سحب واسقاط القوائم بشريط القائمة. فعلى سبيل المثال انقر على File في شريط القائمة ومن ثم تظهر الأوامر الموجودة بهذه القائمة ثم قم بسحب الأمر Open ليصبح فوق الأمر New وهكذا...

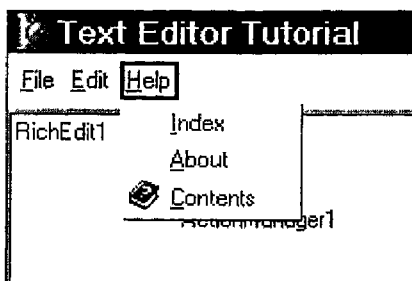


(٤) من قائمة العرض Categories بصندوق الحوار Action Manager اسحب Edit واسقطه في شريط القائمة بالفورمة كما هو موضح بالشكل التالي :



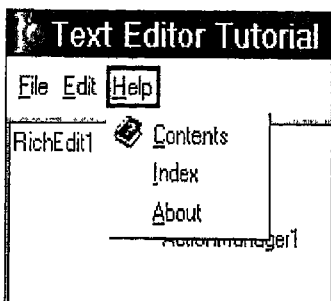
شكل توضيحي :

- (٥) من قائمة العرض Categories بصندوق الحوار Action Manager اسحب Help واسقطه في شريط القائمة بالفورمة كما هو موضح بالشكل التالي :



شكل توضيحي :

- (٦) انقر بالماوس على القائمة Help بشريط القائمة لكي تشاهد محتوياتها ثم اسحب الأمر Contents ليصبح فوق الأمر Index كما هو موضح بالشكل التالي :



شكل توضيحي :

- (٧) اضغط على المفتاح Esc بلوحة المفاتيح أو انقر بالماوس على Help مرة أخرى لغلقتها.



(٨) افتح القائمة File ثم اختر منها الأمر Save All ليتم حفظ التغييرات التي أجريتها بالمشروع.

والآن ستحتاج لأن تضيف شريط أدوات لتوفير طريقة سهلة للوصول للأوامر الموجودة بمجموعة القوائم الموجودة بالمشروع.

إضافة شريط أدوات toolbar للمشروع


بما إنك قمت بإعداد الأفعال في صندوق الحوار Action Manager لذلك تستطيع إضافة بعض من نفس هذه الأفعال والتي كانت مستخدمه بمجموعة القوائم إلى عدد من الأيقونات بشريط أدوات toolbar.

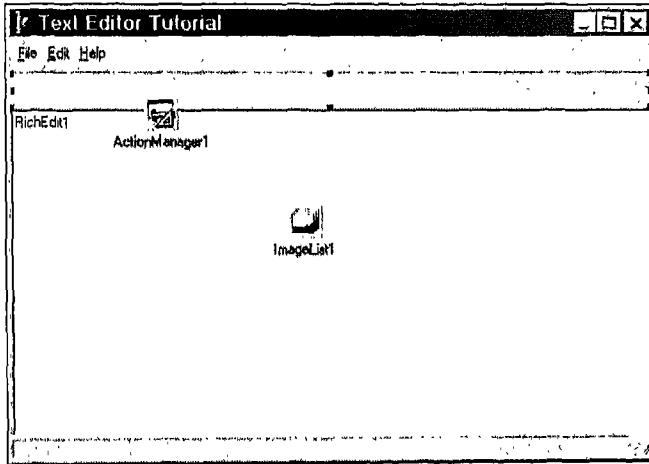
بعد أن تنتهي من شريط الأدوات هذا ستجد أنه مشابهه إلى حد كبير لشريط أدوات برامج الأوفيس ٢٠٠٠.



وللقيام بذلك اتبع الخطوات التالية :

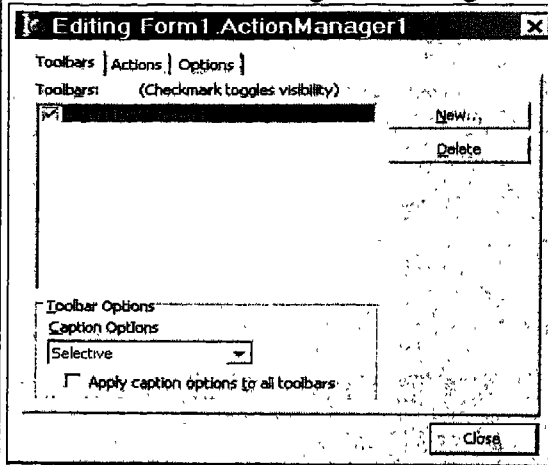
(١) فى الصفحة Additional بالليته المكونات انقر بالماوس نقرا مزدوجا على أيقونة

المكون ActionToolBar  ليتم إضافته إلى الفورمة وهو عبارة عن شريط أدوات فارغ يظهر اسفل شريط القائمة مباشرة كما هو موضح بالشكل التالى :

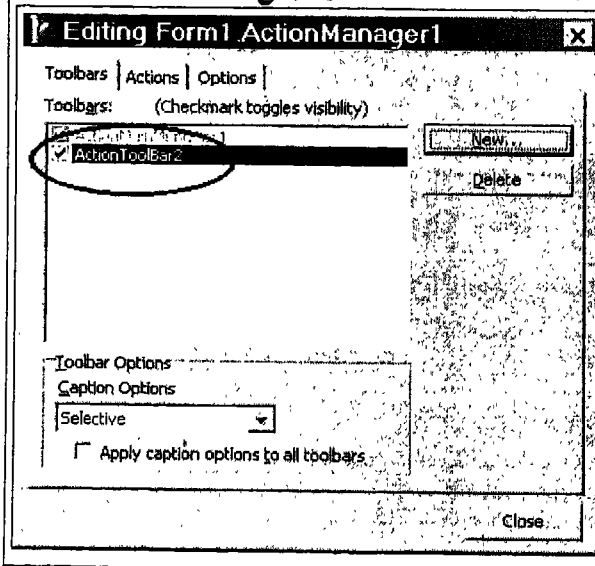


شكل توضيحي :

تستطيع أيضا إضافة شريط أدوات نت النوع action band وذلك عن طريق فتح صندوق الحوار Action Manager وجعل التبويب Toolbars يظهر على السطح كما هو موضح بالشكل التالي :

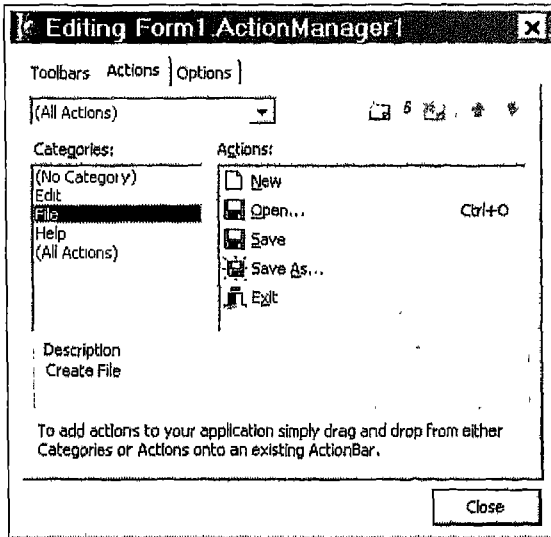


ثم انقر بالماوس على المفتاح New ليتم إضافة شريط أدوات جديد بالفورمة كما يظهر اسم شريط الأدوات الجديد في قائمة العرض Toolbars بصندوق الحوار Action Manager كما هو موضح بالشكل التالي :



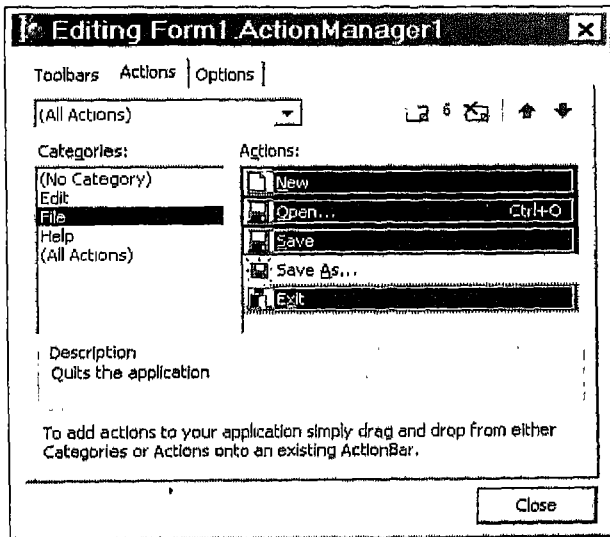


(٢) إذا لم يكن صندوق الحوار Action Manager غير معروض على الشاشة قم بفتحه ثم اختر File بالقائمة Categories كما هو موضح بالشكل التالي :



شكل توضيحي :

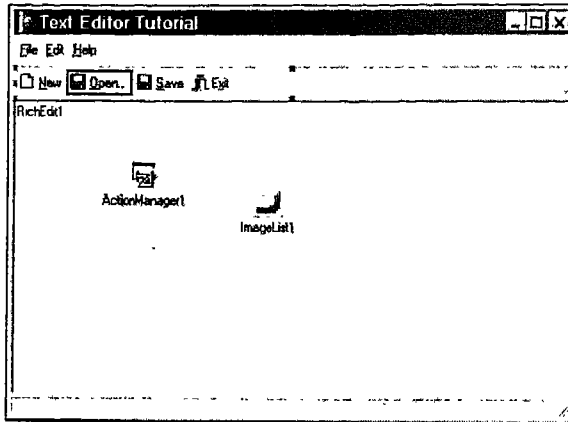
(٣) في القائمة Actions اختر كل من New و Open و Save و Exit كما هو موضح بالشكل التالي :



شكل توضيحي :

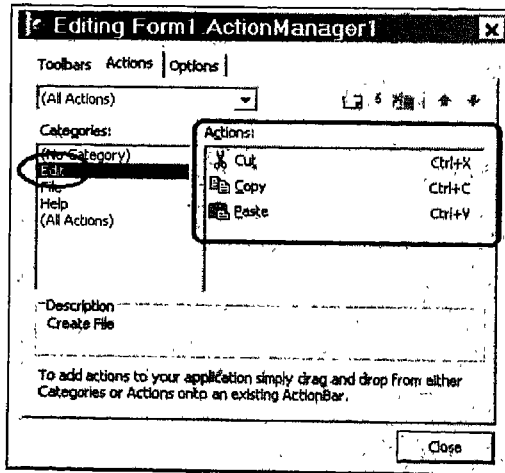


ثم اسحب هذه العناصر بالماوس واسقطهم في شريط الأدوات الموجود بالفورمة لتجد أن هذه العناصر قد ظهرت تلقائيا على شكل أيقونات في شريط الأدوات كما هو موضح بالشكل التالي :



شكل توضيحي :

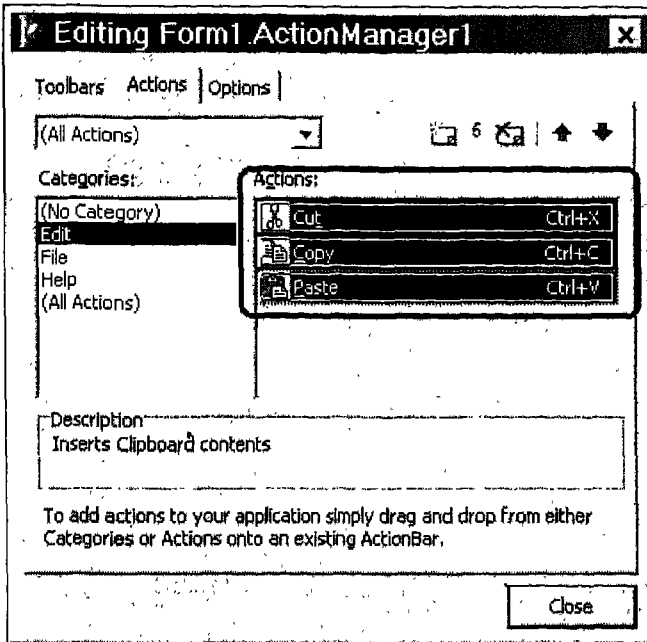
(٤) في صندوق الحوار Action Manager اختر Edit بقائمة العرض Categories لتظهر الأفعال الخاصة بالقسم Edit بقائمة العرض Actions كما هو موضح بالشكل التالي :



شكل توضيحي :

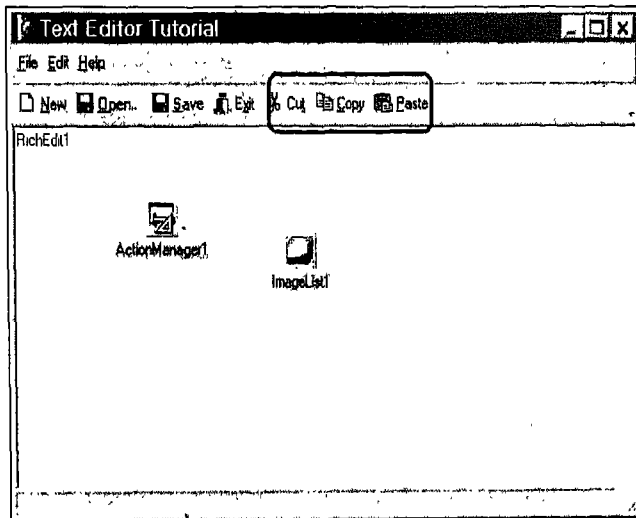
❶ في القائمة Actions اختر كل من Cut و Copy و Paste كما هو موضح بالشكل التالي :

شكل توضيحي :

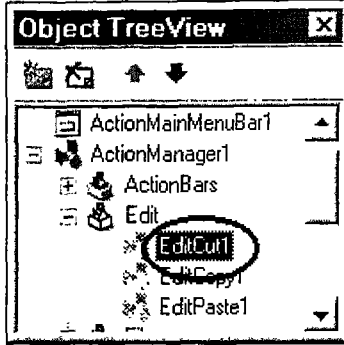


اسقط هذه العناصر في شريط الأدوات بالفورمة لتظهر تلقائيا الأيقونات التي تناظر هذه العناصر كما هو موضح بالشكل التالي :

شكل توضيحي :



لو أنك سحبت الأمر الخطأ واسقطه في شريط الأدوات في هذه الحالة تستطيع أن تسحبه إلى خارج شريط الأدوات مرة أخرى. كما تستطيع أيضا اختيار العنصر في النافذة Object TreeView (كما هو موضح بالشكل التالي) ثم تضغط على المفتاح Delete بلوحة المفاتيح :



شكل توضيحي :

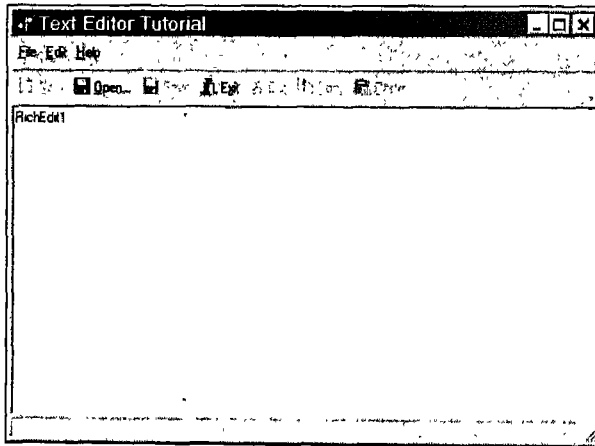
تستطيع أن تعيد ترتيب الأيقونات بشريط الأدوات بأن تسحب أي منهم إلى اليمين أو اليسار.



(٥) افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التي أجريتها.

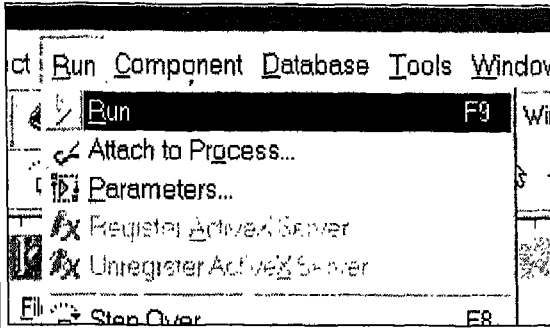
(٦) اضغط على المفتاح F9 بلوحة المفاتيح للبدء في ترجمة وتشغيل المشروع ليظهر

على الشاشة كما هو موضح بالشكل التالي :

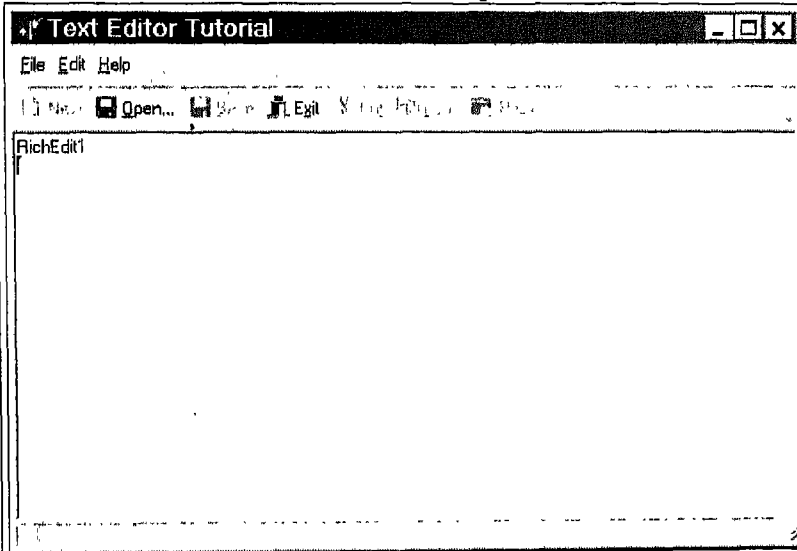


شكل توضيحي :

تستطيع أيضا تشغيل المشروع بالنقر بالماوس على الأيقونة **Run** الموجودة بشريط الأدوات **Debug** أو بفتح القائمة **Run** واختيار **Run** منها كما هو موضح بالشكل التالي :



عندما تقوم بتشغيل المشروع الذي تعده فإن لغة **Delphi** تفتح البرنامج في نافذة مرحلة التشغيل **Run-Time** وهذه النافذة تشبه الفورمة التي قمت بتصميمها كما هو موضح بالشكل التالي :



وستجد أن كل من القوائم وشريط الأدوات يعملان بالرغم من كون بعض الأوامر غير نشطة.

محرر النصوص الذى تعدده أصبح لدية بالفعل العديد من الأدوات الوظيفية. وأنت تستطيع كتابة ما يحلو لك فى منطقة النص كما تستطيع أيضا القيام بكل من عمليات القص والنسخ و اللصق من خلال الأيقونات المخصصة لها بشريط الأدوات. ولكن على كل حال لايزال هناك الكثير الذى ينبغى عمله لتنشيط وتفعيل الأوامر الموجودة بالقوائم.

(٧) لكى تعود مرة أخرى لمود التصميم Design-Time انقر بالماوس على الزر (X) الموجود بالركن الأيمن العلوى لنافذة التطبيق.


إلغاء منطقة النص من محتوياتها

عندما قمت بتشغيل البرنامج الذى تعدده وجدت أن الاسم RichEdit1 قد ظهر فى منطقة النص. وأنت تستطيع إزالة هذا النص باستخدام Strings List Editor. ولو أنك لم تقم بتنظيف النص الآن فإن النص ينبغى مسحه عند الإعلان الابتدائى initializing الفورمة الأساسية فى آخر خطوة.

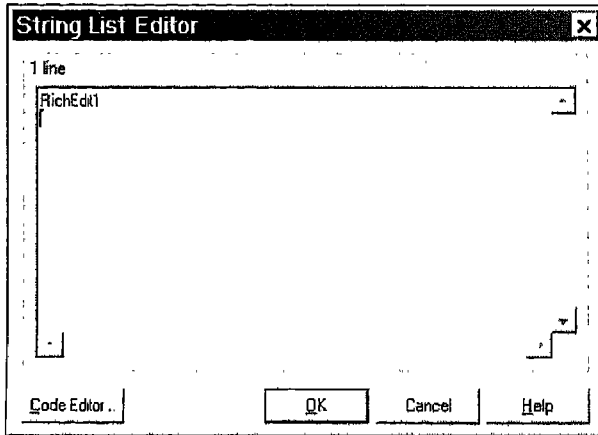
لتنظيف منطقة النص اتبع الخطوات التالية :

(١) فى الفورمة الأساسية انقر بالماوس على المكون RichEdit1.

(٢) فى نافذة Object Inspector وعند الخاصية Lines انقر بالماوس على

الأيقونة  المجاورة للقيمة TStrings ليظهر على الشاشة المحرر String

List Editor الموضح فى الشكل التالى :

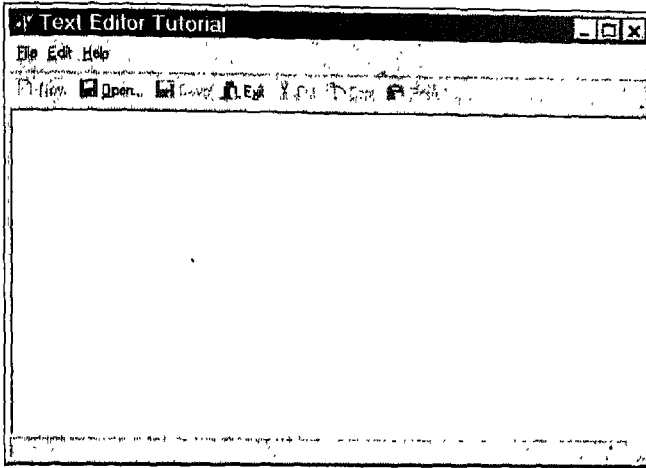


شكل توضيحي :

(٣) اختر النص RichEdit1 والذي ترغب فى مسحه ثم قم بإزالته من المحرر Filter ثم اضغط على المفتاح Delete بلوحة المفاتيح ثم انقر بالماوس على المفتاح Ok.

(٤) افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التى أجريتها ثم حاول تشغيل البرنامج مرة أخرى.

تلاحظ الآن أن منطقة تحرير النص أصبحت نظيفة تماما وذلك عند عرض الفورمة الأساسية على الشاشة كما هو موضح بالشكل التالى :



شكل توضيحي :

كتابة أدوات معالجة الأحداث Event handlers

حتى هذه المرحلة قمت بإعداد التطبيق الخاص بك بدون كتابة أى كود برمجى ولو حتى سطر واحد فقط. هذا وعن طريق استخدام Object Inspector لتحديد قيم الخصائص فى مرحلة التصميم Design-Time فإنك إذن قد استفدت تمام الاستفادة من بيئة التطوير RAD الخاصة بلغة Delphi.

فى هذا الجزء من الفصل سنقوم سويا بكتابة الإجراءات التى يطلق عليها أدوات معالجة الأحداث Event handlers والتى تستجيب لمدخلات المستخدم فى أثناء كون التطبيق فى مرحلة التشغيل Run-Time. وستقوم أيضا بإقامة الاتصال أو الربط بين أدوات

معاملة الأحداث Event handlers هذه والعناصر الموجودة في القوائم وشريط الأدوات ومن ثم عندما يتم اختيار أى عنصر يقوم التطبيق الذى تعده على الفور بتنفيذ الكود البرمجى الموجود بأداة معاملة الحدث الخاصة بهذا العنصر.

بالنسبة للأفعال الغير قياسية ينبغى عليك إنشاء أداة معاملة حدث. أما بالنسبة للأفعال القياسية مثل الفعل الخاص بالأمر Exit بالقائمة File والفعل الخاص بالأمر Paste بالقائمة Edit فإن الأحداث المرتبطة بها تكون داخل الكود البرمجى الذى يتم تكوينه تلقائيا لهذه النوعية من الأفعال. ولكن على كل حال يمكن القول بأنه بالنسبة لبعض الأفعال القياسية مثل الفعل الخاص بالأمر Save As بالقائمة File ستحتاج لأن تكتب أداة حدث خاصة بهذا الفعل (القياسى) وذلك من أجل تفصيل طريقة عمل هذا الأمر.

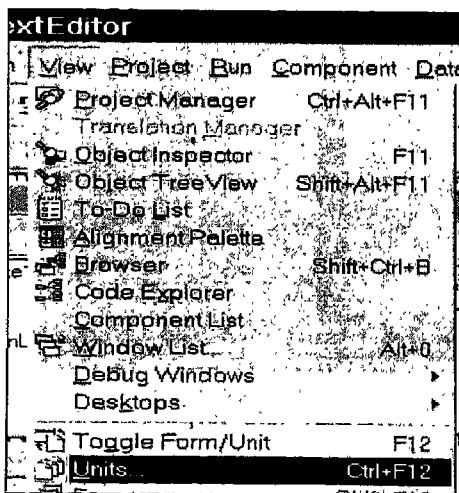
بسبب أن عناصر القوائم وشريط الأدوات كلها متركزة فى صندوق الحوار Action Manager لذلك تستطيع أن تنشأ أدوات معاملة الأحداث Event handlers من نفس صندوق الحوار هذا.



لكى تنشأ أداة معاملة الحدث المرتبط بالأمر New (بالقائمة File) اتبع الخطوات

التالية :

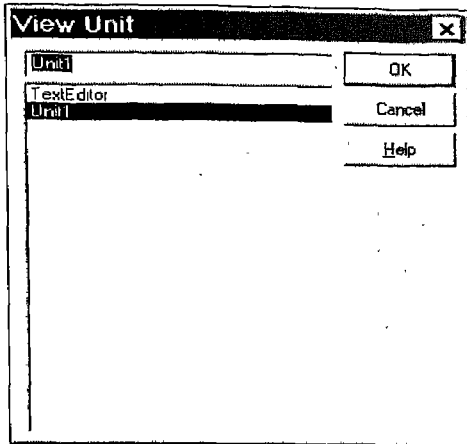
(١) افتح القائمة View واختر منها Units كما هو موضح بالشكل التالى :



شكل توضيحي :



ليظهر على الشاشة صندوق الحوار View Unit الموضح في الشكل التالي :

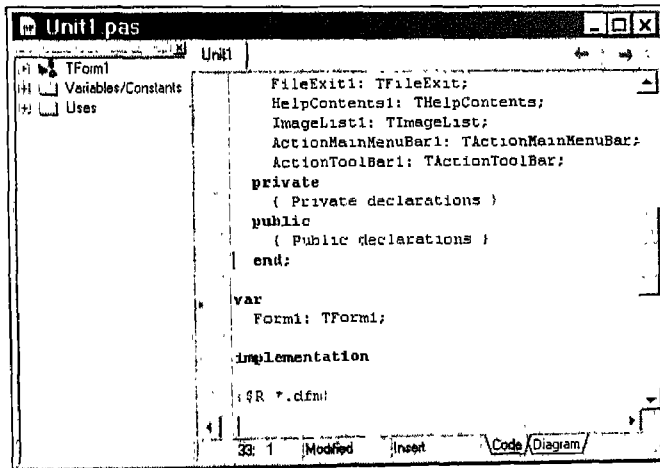


شكل توضيحي :

يمكن فتح صندوق الحوار View Unit بطريقة أخرى وهي الضغط على المفاتيح Ctrl+F12 بلوحة المفاتيح.



بصندوق الحوار View Unit اختر الوحدة Unit1 ثم انقر بالماوس على المفتاح Ok وذلك لعرض الكود البرمجي الملحق بالفورمة Form1 كما هو موضح بالشكل التالي :



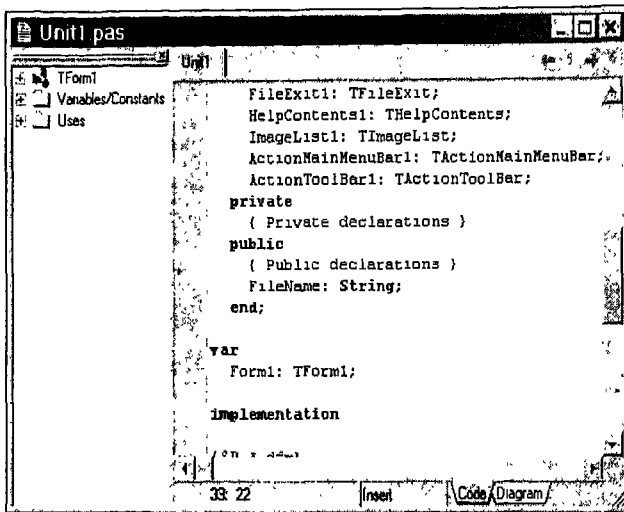
شكل توضيحي :

(٢) ستحتاج لأن تعلن عن اسم الملف الذي سيتم استخدامه في أداة معاملة الحدث كما ستحتاج أيضا إلى إضافة خاصية مفصلة لاسم الملف هذا وذلك لجعل من

الممكن الوصول إليه من أى منطقة بالتطبيق. والآن وفي بداية ملف الوحدة Unit1.pas ابحث عن موضع المقطع الذى يضم مجموعة جمل الإعلانات العامة الخاصة بالقطاع TForm1 وفى السطر الذى يلي السطر {Public declarations} اكتب الآتى :

FileName: String;

كما هو موضح بالشكل التالى :



شكل توضيحي :

(٣) اضغط على المفتاح F12 بلوحة المفاتيح للعودة مرة أخرى إلى الفورمة الأساسية.

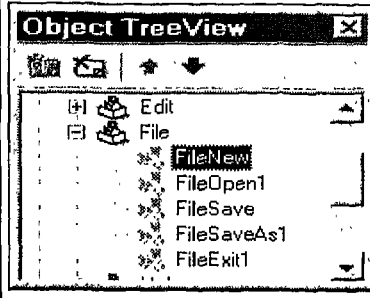
من خلال المفتاح F12 يمكن أن تتحرك ذهابا وإيابا بين الفورمة والكود البرمجى المرتبط بها.



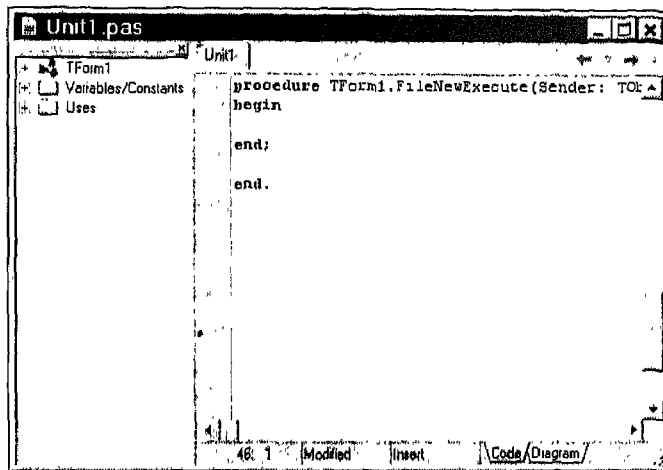
(٤) انقر بالماوس نقر مزدوجا على المكون ActionManager بالفورمة لفتح صندوق الحوار Action Manager.

(٥) فى صندوق الحوار Action Manager اختر القسم File بالقائمة Categories وبعد ذلك انقر بالماوس نقر مزدوجا على الأيقونة New action.

تستطيع أيضا أن تفقر بالماوس نقرا مزدوجا على الفعل FileNew الموجود بالقسم File وذلك في النافذة Object TreeView كما هو موضح بالشكل التالي :



يتم فتح محرر الكود البرمجي Code Editor وستجد أن مؤشر الكتابة داخل أداة معالجة الحدث الخاص بهذا الفعل كما هو موضح بالشكل التالي :



شكل توضيحي :

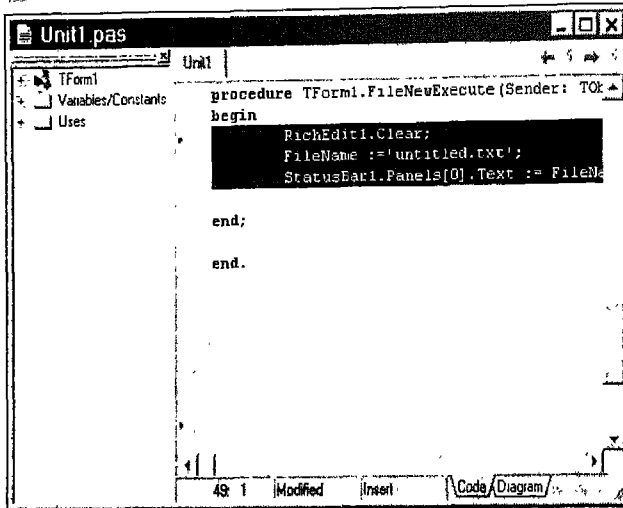
(٦) ضع مؤشر الكتابة بمحرر الكود البرمجي (بين Begin و End) ثم اكتب السطور التالية :

```

RichEdit1.Clear;
FileName := 'untitled.txt';
StatusBar1.Panels[0].Text := FileName;
    
```

ليصبح الكود البرمجي الخاص بهذا الحدث كما هو موضح بالشكل التالي :

شكل توضيحي :



(٧) افتح القائمة File ثم اختر منها الأمر Save All ليتم حفظ التغييرات التي أجريتها الآن.

تستطيع تغيير حجم نافذة محرر الكود البرمجي لكي تقلل من الحاجة لاستخدام شريط الحركة الأفقي.

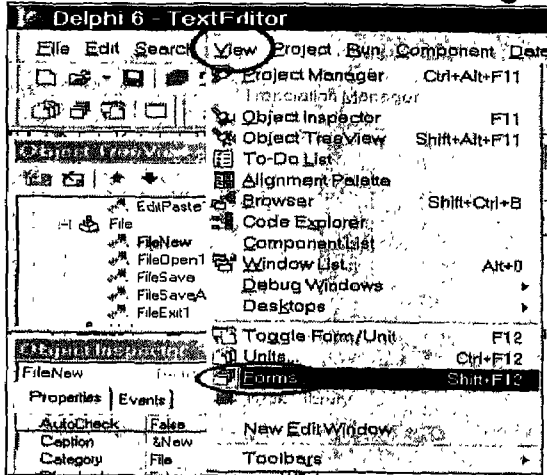


إنشاء أداة معاملة الحدث المرتبط بالأمر Open

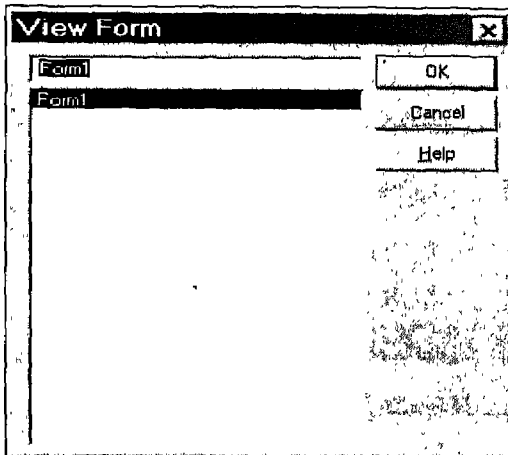
لكي يتم فتح ملف في محرر النصوص الذي تعدّه فإنك تحتاج لأن يظهر على الشاشة صندوق الحوار Open (وهو من ضمن صناديق الحوار القياسية بيئة الويندوز). ولقد قممت بالفعل بإضافة الأمر القياسي Open إلى القائمة File وذلك بصندوق الحوار Action Manager والذي يتضمن تلقائياً صندوق الحوار القياسي Open. ولكن على كل حال ستظل في حاجة لتفصيل أداة الحدث المرتبط بهذا الأمر. وللقيام بذلك اتبع الخطوات التالية :

(١) اضغط على المفتاح F12 بلوحة المفاتيح للذهاب إلى الفورمة الأساسية بمود التصميم.

يمكن القيام بنفس المهمة عن طريق فتح القائمة View ثم اختيار الأمر Forms منها كما هو موضح بالشكل التالي :



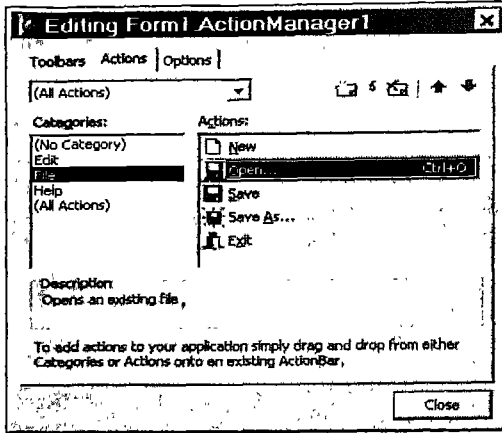
ليظهر على الشاشة صندوق الحوار View form الموضح في الشكل التالي :



ثم التعليم على Form1 ثم النقر بالماوس على المفتاح Ok.

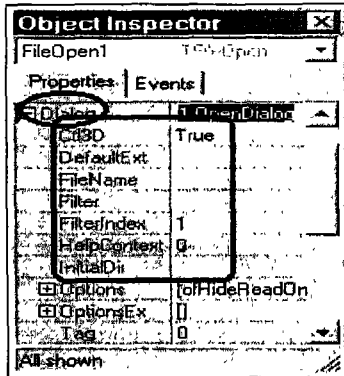
يمكن أيضا فتح صندوق الحوار View Form عن طريق الضغط على المفاتيح Shift+F12.

(٢) انقر بالماوس نقرًا مزدوجًا على أيقونة Action Manager لفتح صندوق الحوار Action Manager م اختر الفعل Open بالقسم File كما هو موضح بالشكل التالي :



شكل توضيحي :

(٣) في النافذة Object Inspector انقر بالماوس على العلامة (+) الموجودة على يسار الخاصية Dialog لعرض الخصائص الفرعية لهذه الخاصية الأساسية كما هو موضح بالشكل التالي :

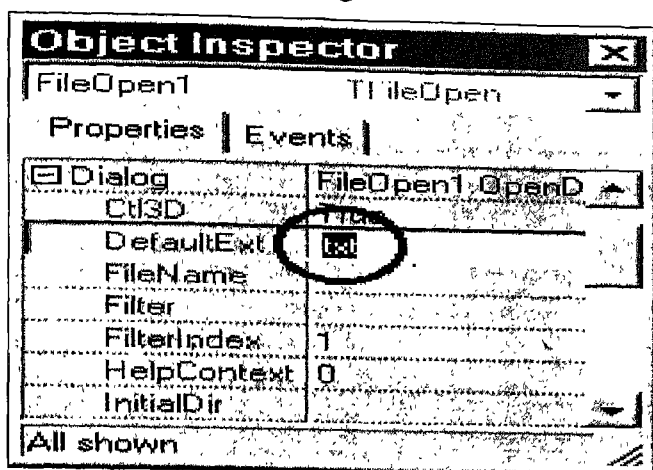


شكل توضيحي :

تقوم لغة Delphi بتسمية الخاصية OpenDialog للفعل FileOpen1 وهذه التسمية تعتبر تسمية افتراضية. وعندما يتم استدعاء الأمر Execute الخاص بالخاصية OpenDialog1 فإنه يتم فتح صندوق الحوار الأساسي Open لفتح الملفات.

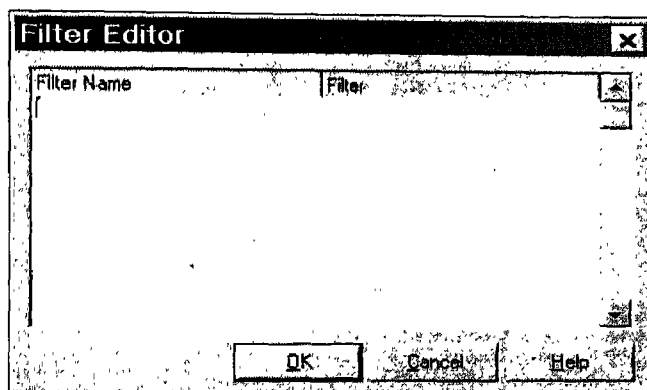


- (٤) قم بتحديد الخصائص التالية الخاصة بـ FileOpen1.Dialog :
- عند الخاصية DefaultExt اكتب txt كما هو موضح بالشكل التالي :



شكل توضيحي :

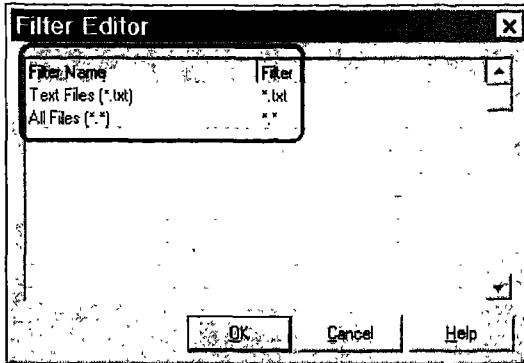
- انقر بالماوس انقر مزدوجا على الحقل المجاور للخاصية Filter (او انقر بالماوس على الأيقونة لت) ليظهر على الشاشة صندوق الحوار Filter Editor الموضح في الشكل التالي :



شكل توضيحي :

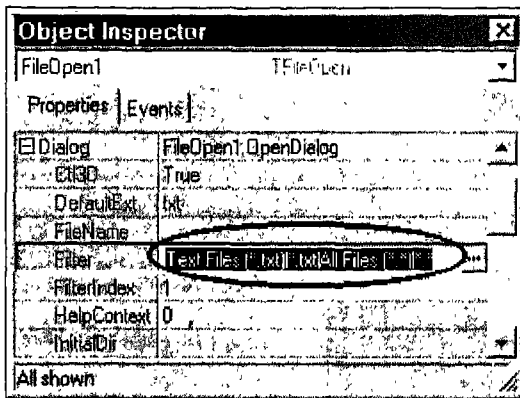
في الخلية الموجودة بالصف الأول والعمود Filter Name اكتب Text files

(*.txt) ثم في الخلية الموجودة بالصف الأول والعمود Filter اكتب *.txt. ثم في الخلية الموجودة بالصف الثاني والعمود Filter Name اكتب All Files (*.*) ثم في الخلية الموجودة بالصف الثاني والعمود Filter اكتب *.* كما هو موضح بالشكل التالي :



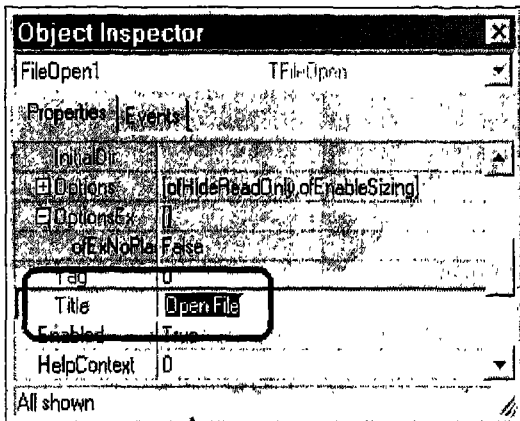
شكل توضيحي :

والآن انقر بالماوس على المفتاح OK لغلق صندوق الحوار. والآن ستجد القيمة المخصصة للخاصية Filter كما هو موضح بالشكل التالي :



شكل توضيحي :

عند الخاصية Title اكتب Open File كما هو موضح بالشكل التالي :

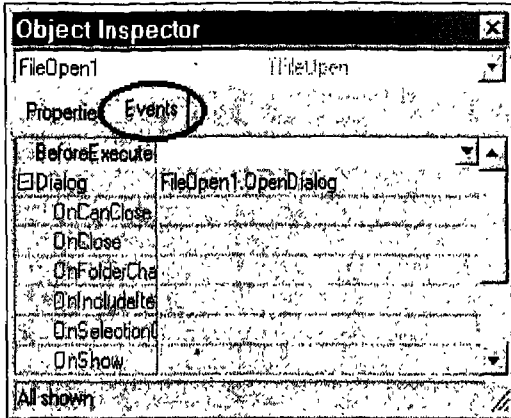


شكل توضيحي :

كلمة Open File ستظهر كعنوان لمندوق الحوار القياسي Open.

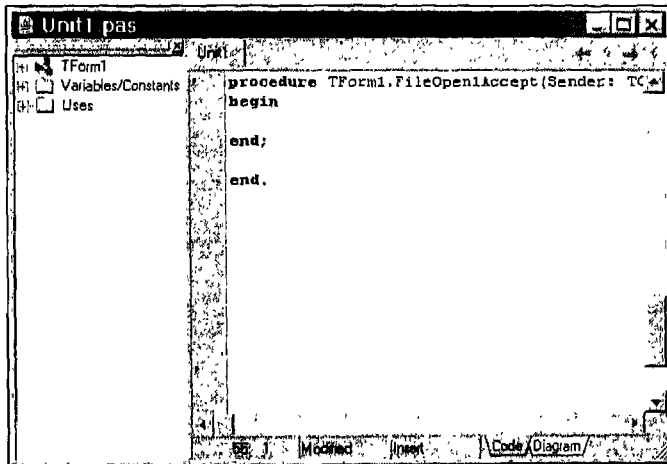


(٥) انقر بالماوس على التبويب Events ليظهر على السطح بالنافذة Object Inspector كما هو موضح بالشكل التالي :



شكل توضيحي :

بهذا التبويب انقر بالماوس نقرا مزدوجا على الحدث OnAccept ليظهر على الشاشة محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالي :



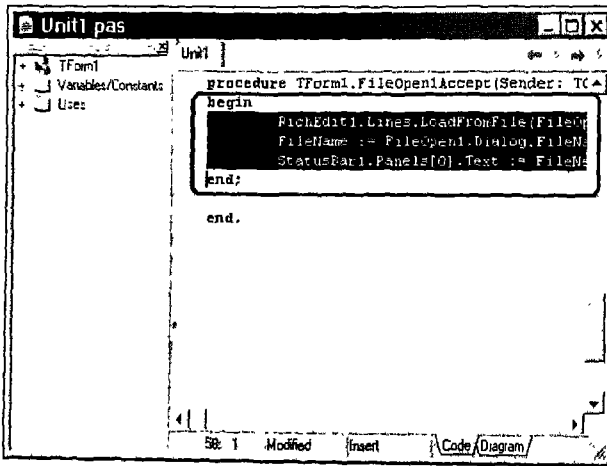
شكل توضيحي :

(٦) كما هو موضح بالشكل السابق تلاحظ أن مؤشر الكتابة موجود داخل أداة المعاملة لهذا الحدث.

(٧) تحرك قليلا إلى جهة اليمين ثم اكتب مجموعة السطور التالية :

```
RichEdit1.Lines.LoadFromFile(FileOpen1.Dialog.FileName);
FileName := FileOpen1.Dialog.FileName;
StatusBar1.Panels[0].Text := FileName;
```

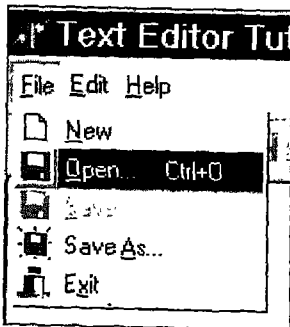
ومن ثم يصبح محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالي :



شكل توضيحي :

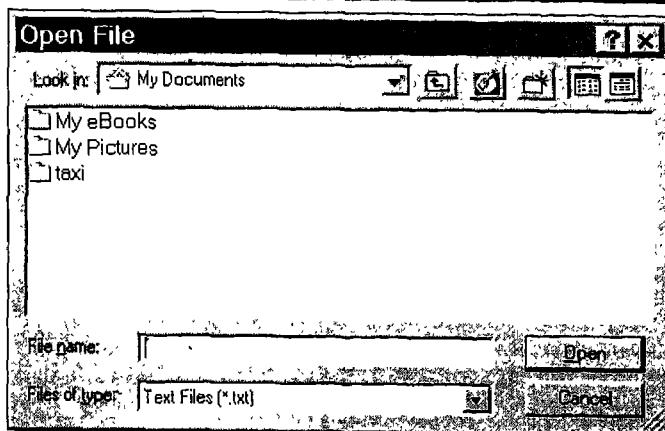
(٨) قم الآن بتشغيل التطبيق بالضغط على المفتاح F9 بلوحة المفاتيح ثم في فورمة

التطبيق الأساسية انقر بالماوس على الأيقونة Open أو اضغط على المفتاحين Ctrl+O بلوحة المفاتيح أو افتح القائمة File ثم اختر منها الأمر Open كما هو موضح بالشكل التالي :



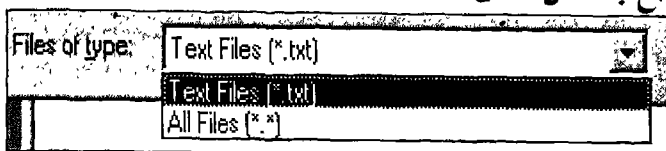
شكل توضيحي :

(٩) ليظهر على الشاشة صندوق الحوار Open File الموضح في الشكل التالي :



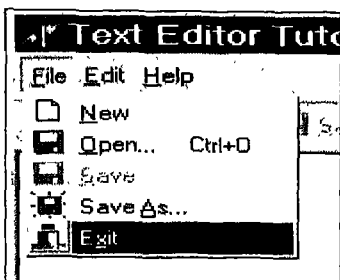
شكل توضيحي :

عندما تفتح القائمة المنسدلة Files of type: تلاحظ أنها تشتمل على كل من الاختيار Text Files (*.txt) والاختيار All Files (*.*) كما هو موضح بالشكل التالي :



وهذا نتيجة ما قمنا به في الخطوة رقم (٤).

- (١٠) انقر بالماوس على المفتاح Cancel لإغلاق صندوق الحوار Open File ثم افتح القائمة File واختر منها المر Exit كما هو موضح بالشكل التالي لإنهاء البرنامج :



شكل توضيحي :

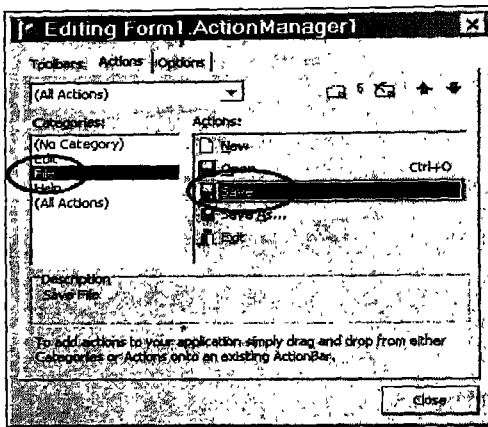
- (١١) افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التي أجريتها بالمشروع.

إنشاء أداة معاملة الحدث للأمر Save

لكي تنشأ أداة معاملة الحدث الخاص بالأمر Save اتبع الخطوات التالية :

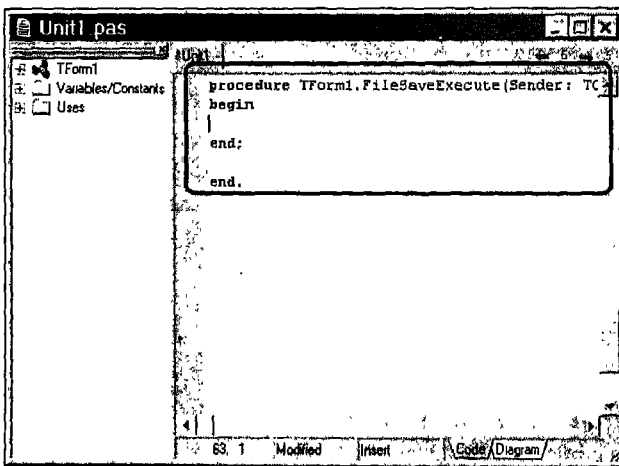
(١) اضغط على المفتاح F12 بلوحة المفاتيح لعرض الفورمة على الشاشة. ثم انقر بالماوس نقرا مزدوجا على أيقونة المكون ActionManager بالفورمة لفتح صندوق الحوار Action Manager (إذا كان مغلقا).

(٢) انقر بالماوس نقرا مزدوجا على الفعل Save بالقسم File الموضح في الشكل التالي :



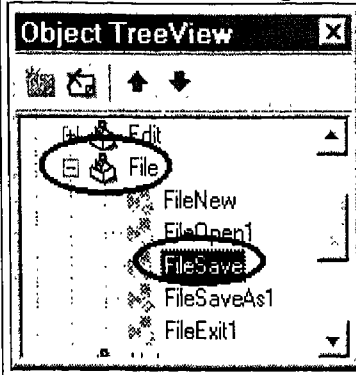
شكل توضيحي :

ليتم فتح محرر الكود البرمجي Code Editor وستجد أن مؤشر الكتابة موجود داخل أداة معاملة الحدث الخاص بالفعل Save كما هو موضح بالشكل التالي :



شكل توضيحي :

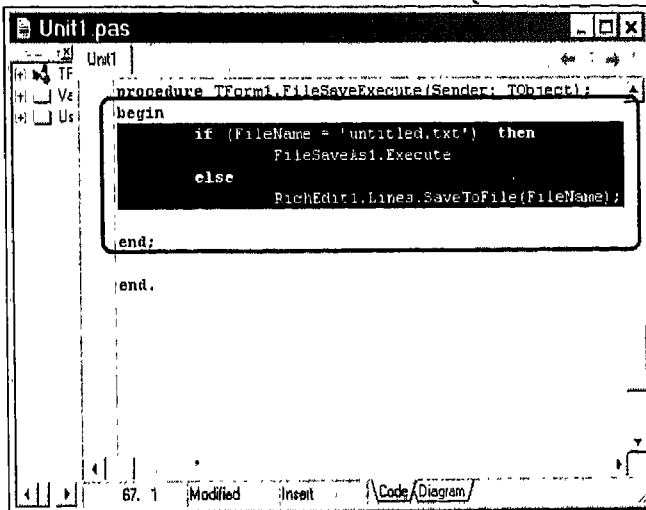
تستطيع أيضا النقر بالماوس نقرا مزدوجا على الفعل FileSave بالقسم File بنافذة Object TreeView كما هو موضح بالشكل التالي :



(٣) تحرك بمؤشر الكتابة إلى اليمين قليلا بالسطر الفارغ بين End و Begin ثم اكتب السطور التالية :

```
if (FileName = 'untitled.txt') then
    FileSaveAs1.Execute
else
    RichEdit1.Lines.SaveToFile(FileName);
```

كما هو موضح بالشكل التالي :



شكل توضيحي :

هذا الكود البرمجي يخبر محرر النصوص بأن يعرض صندوق الحوار Save As إذا كان الملف المطلوب حفظه لم يتم تسميته بعد ومن ثم يستطيع المستخدم تخصيص اسم له من خلال صندوق الحوار Save As. ولكن في حالة وجود اسم للملف فإنه يتم حفظه بنفس هذا الاسم.



الخاصية BeforeExecute للفعل FileSaveAs1 تقوم تلقائياً بتكوين اسم للأمر Save As وهذا لا يحدث بالنسبة للأمر Save بالقائمة File.



والآن افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التي أجريتها بالمشروع.

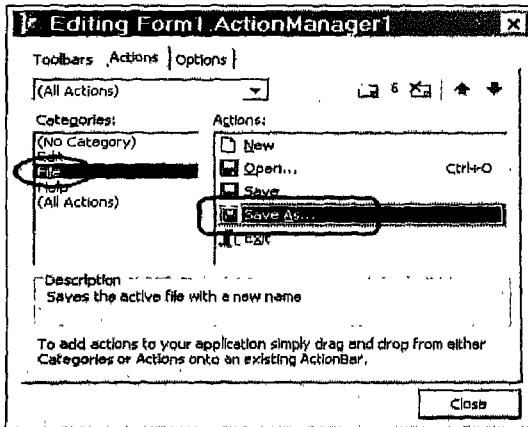
إنشاء أداة معاملة الحدث المرتبط للأمر Save As

عندما يتم استدعاء الأسلوب Execute الخاص بـ SaveDialog يظهر على الشاشة صندوق الحوار القياسي Save As لحفظ الملفات. هذا ولكي تنشأ أداة معاملة الحدث المرتبط للأمر Save As اتبع الخطوات التالية :

(١) اضغط على المفتاح F12 بلوحة المفاتيح لعرض الفورمة على الشاشة. ثم انقر

بالماوس نقرا مزدوجا على أيقونة المكون ActionManager بالفورمة لفتح صندوق الحوار Action Manager (إذا كان مغلقاً).

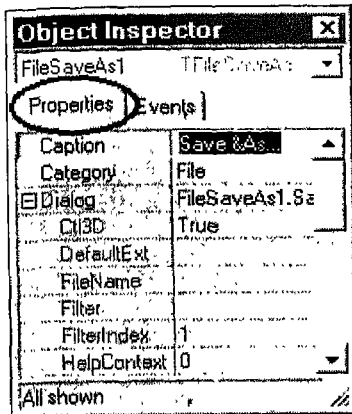
(٢) اختر الفعل File SaveAs كما هو موضح بالشكل التالي :



شكل توضيحي :

(٣) في النافذة Object Inspector انقر بالماوس على التبويب Properties كما هو

موضح بالشكل التالي :

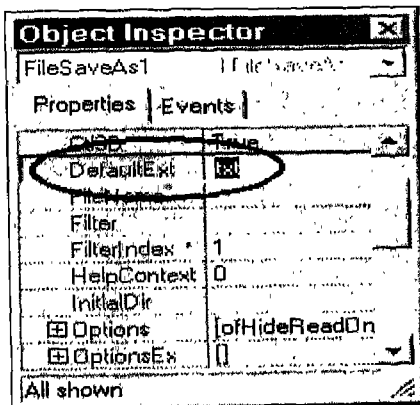


شكل توضيحي :

قم بتحديد قيم الخصائص التالية لصندوق الحوار FileSaveAs1 (الذي تخصص له لغة Delphi الاسم الافتراضي FileSaveAs1).

(٤) انقر بالماوس على العلامة (+) الموجودة على يسار الخاصية Dialog ثم قم بتحديد قيم الخصائص التالية :

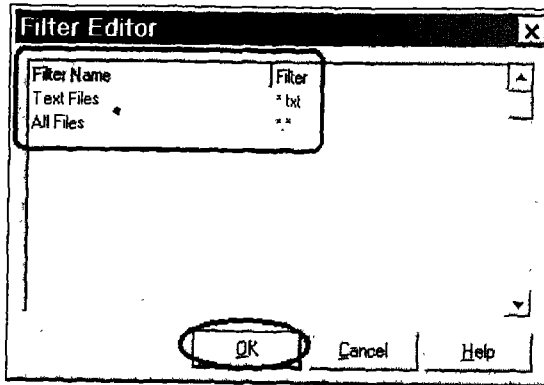
عند الخاصية DefaultExt اكتب txt كما هو موضح بالشكل التالي :



شكل توضيحي :

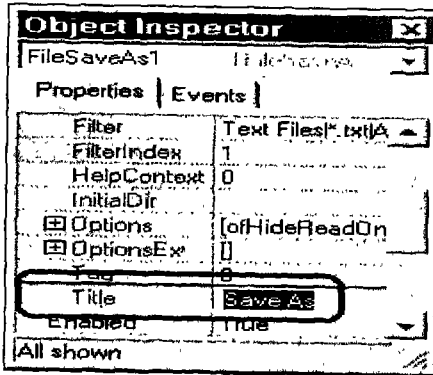
انقر بالماوس نقرًا مزدوجًا على الحقل المجاور للخاصية Filter لعرض صندوق الحوار Filter Editor والذي يتم من خلاله توصيف الفلاتر الخاصة بأنواع الملفات

مثلاً حدث بالنسبة لصندوق الحوار Open. والآن وفي الصف الأول وبالعمود Filter Name اكتب Text Files (*.txt) ثم في العمود Filter اكتب *.txt. أما في الصف الثاني وبالعمود Filter Name اكتب All Files (*.*) ثم في العمود Filter اكتب *.* ثم انقر بالماوس على المفتاح Ok كما هو موضح بالشكل التالي :



شكل توضيحي :

عند الخاصية Title اكتب Save As كما هو موضح بالشكل التالي :



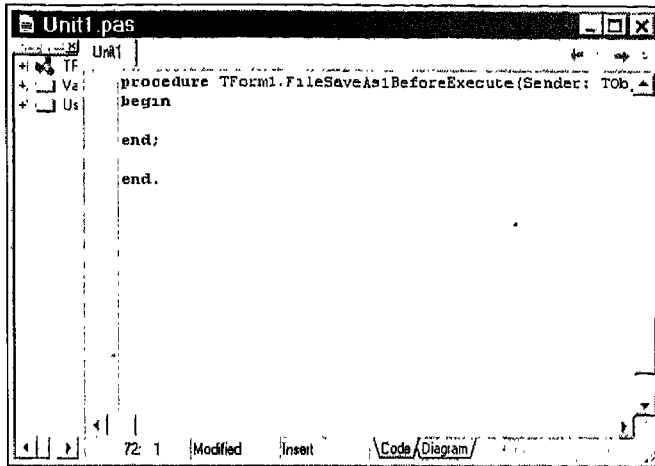
شكل توضيحي :

القيمة التي يتم تخصيصها للخاصية Title تظهر بشرط العنوان لصندوق الحوار Save As كما سنرى بعد ذلك.



(٥) في النافذة Object Inspector انقر بالماوس على التبويب Events ليظهر على السطح وفيه انقر بالماوس نقرا مزدوجا على الحقل المجاور للخاصية BeforeExecute ليظهر هيكل الكود البرمجي الخاص بـ

Code Editor والذي فيه نشاهد مؤشر الكتابة داخل الكود البرمجي لأداة معاملة الحدث للأمر SaveAs كما هو موضح بالشكل التالي :

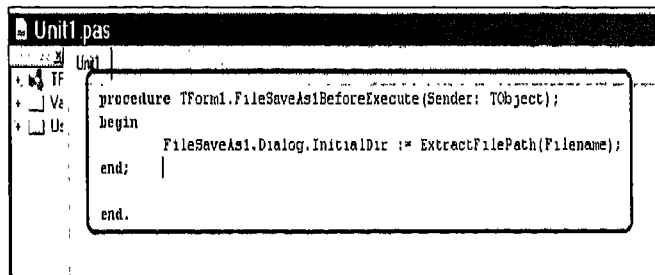


شكل توضيحي :

(٦) بالسطر الموجود به مؤشر الكتابة بمحرر الكود البرمجي اكتب الكود الآتي :

```
FileSaveAs1.Dialog.InitialDir := ExtractFilePath(Filename);
```

كما هو موضح بالشكل التالي :

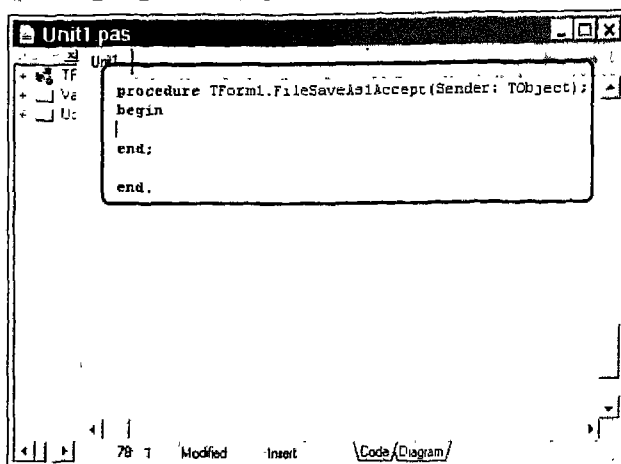


شكل توضيحي :

(٧) في النافذة Object Inspector لابد أن يكون التبويب Events لا يزال موجود على السطح. وفيه انقر بالماوس نقرا مزدوجا على الحقل المجاور للحدث OnAccept لكي يظهر هيكل الكود البرمجي الخاص بـ FileSaveAs1Accept وذلك في محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالي :



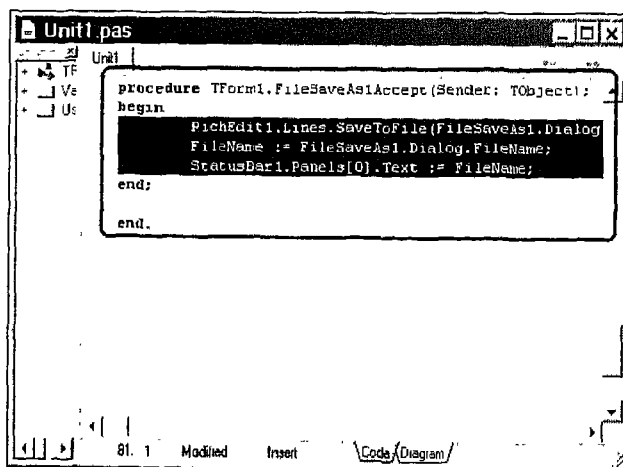
شكل توضيحي :



(٨) بالسطر الموجود به مؤشر الكتابة اكتب الكود التالي :

```
RichEdit1.Lines.SaveToFile(FileSaveAs1.Dialog.FileName);  
FileName := FileSaveAs1.Dialog.FileName;  
StatusBar1.Panels[0].Text := FileName;
```

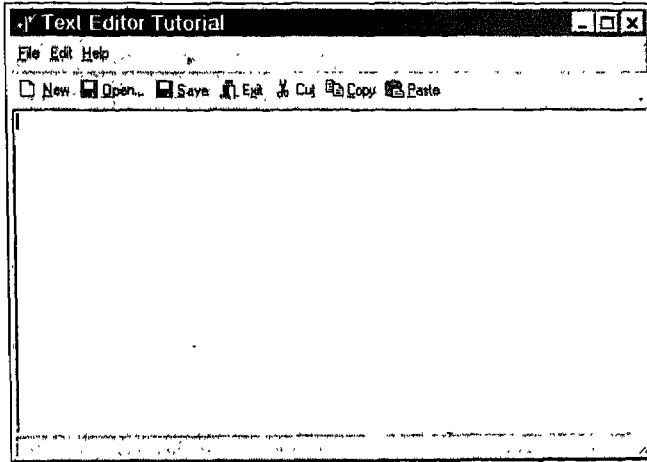
كما هو موضح بالشكل التالي :



شكل توضيحي :

(٩) افتح القائمة File واختر منها الأمر Save All لحفظ كافة التغييرات التي أجريتها بالمشروع.

(١٠) لكى تشاهد الحالة التى وصل إليها التطبيق قم بتشغيله بالضغط على المفتاح F9 بلوحة المفاتيح لتظهر على الشاشة الفورمة الأساسية لهذا التطبيق كما هو موضح بالشكل التالى :

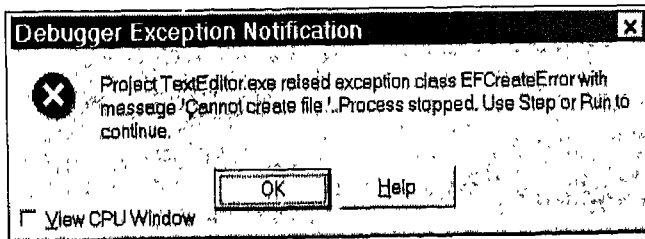


شكل توضيحي :

تلاحظ أن أغلب الأوامر الموجودة بالقوائم وكذلك أغلب الأيقونات الموجودة بشرط الأدوات أصبحت تعمل الآن بالرغم من أنك لم تنتهى بعد من المشروع.



لو أنك تشاهد أى رسائل خطأ عند النقر بالماوس على أى أيقونة من أيقونات المشروع مثل رسالة الخطأ الموضحة فى الشكل التالى :

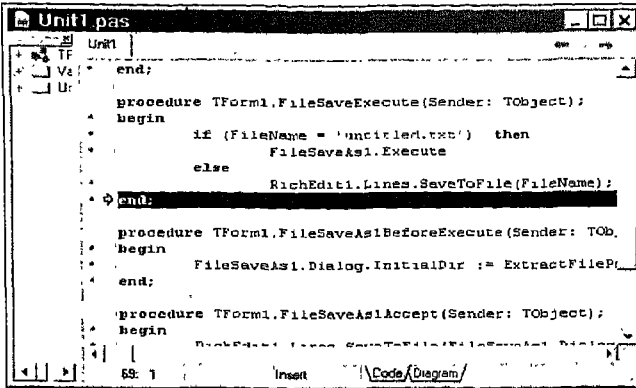


شكل توضيحي :

فى هذه الحالة انقر بالماوس على المفتاح Ok لكى تذهب إلى موضع الخطأ فى الكود البرمجى بنافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :



شكل توضيحي :



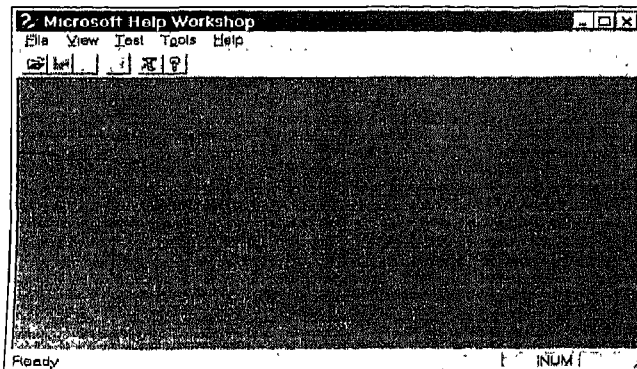
وبهذه الطريقة تستطيع إصلاح الخطأ الذي حدث بالكود البرمجي. وعلى العموم عليك أن تتأكد من أنك اتبعت بطريقة صحيحة الخطوات التي ذكرناها قبل ذلك. (١١) للعودة مرة أخرى إلى مود التصميم انقر بالماوس على الزر (X) بالركن الأيمن العلوي لنافذة التطبيق.

(١٢) افتح القائمة File واختر منها الأمر Save All (او اضغط على مجموعة المفاتيح Shift+Ctrl+S بلوحة المفاتيح) لحفظ التغييرات التي أجريتها بالمشروع.

إنشاء ملف مساعدة

إنها لفكرة جيدة أن يتم إنشاء ملف مساعدة يشرح كيفية استخدام التطبيق الذي تعده. وفي هذا الصدد نقول إن لغة Delphi تعمل على توفير ورشة العمل Microsoft Help Workshop والموجودة في المسار C:\Program Files\Borland\Delphi6\Help\Tools والموضحة في الشكل التالي :

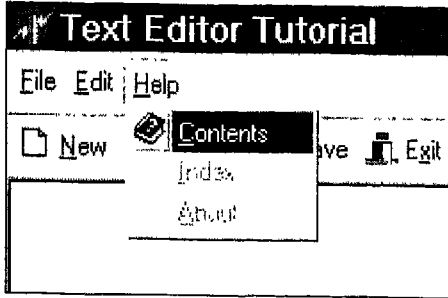
شكل توضيحي :





وورشة العمل هذه تتضمن معلومات عن تصميم وترجمة compiling أى ملف من ملفات المساعدة التى تعمل ببيئة الويندوز.

هذا وفى تطبيق محرر النصوص الذى نحن بصدده هنا نود أن نجعل المستخدمين له لديهم القدرة على الوصول إلى ملف المساعدة (المشتمل على التبويب Contents والتبويب Index) عن طريق الأمر Contents بقائمة Help كما هو موضح بالشكل التالى :



شكل توضيحي :

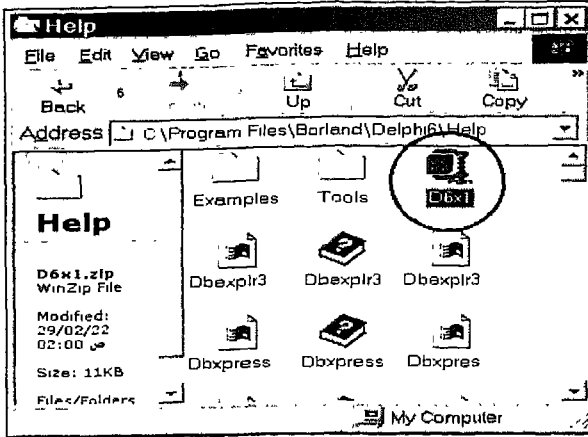
فيما سبق قمنا بإنشاء كل من الفعل HelpContents والفعل HelpIndex فى صندوق الحوار Action Manager وذلك بهدف عرض التبويب Contents أو التبويب Index بملف المساعدة المترجم. والآن أنت فى حاجة لأن تخصص قيم ثابتة لمعاملات المساعدة كما تحتاج أيضا لإنشاء أدوات معاملة الأحداث التى تعمل على عرض ما يرغبه المستخدم من الموضوعات الموجودة بنظام المساعدة.

لكى تستخدم أوامر المساعدة عليك أن تقوم بإنشاء وترجمة ملف مساعدة يعمل ببيئة الويندوز. ونود هنا القول بأن إنشاء ملفات المساعدة موضوع كبير إلى حد ما ولا يمكننا تناوله بالتفصيل فى هذا الفصل. ولكن على العموم تستطيع تحميل كل من ملف النموذج المسمى TextEditor.rtf وملف المساعدة المسمى TextEditor.hlp وملف المحتويات المسمى TextEditor.cnt وذلك عن طريق الخطوات التالية :

(١) من خلال المسار C:\Project Files\Borland\Delphi6\Help افتح الملف

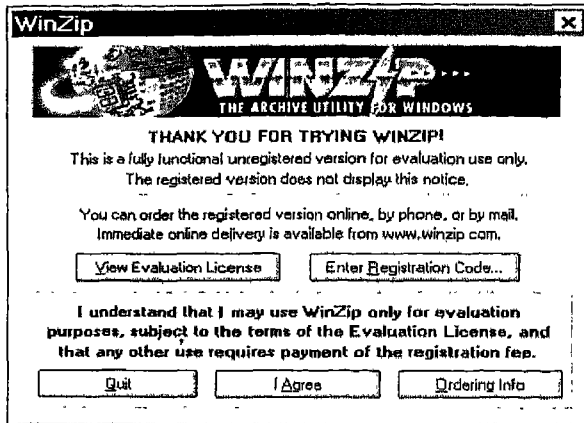
D6X1.zip الموضح فى الشكل التالى :

شكل توضيحي :



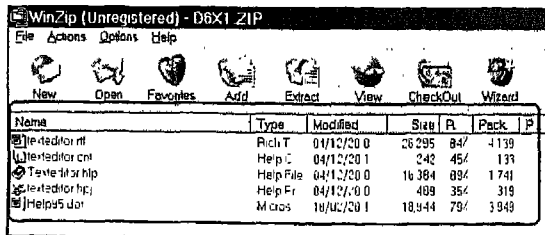
(٢) يظهر الآن على الشاشة صندوق الحوار الافتتاحي للبرنامج WinZip كما هو موضح بالشكل التالي :

شكل توضيحي :

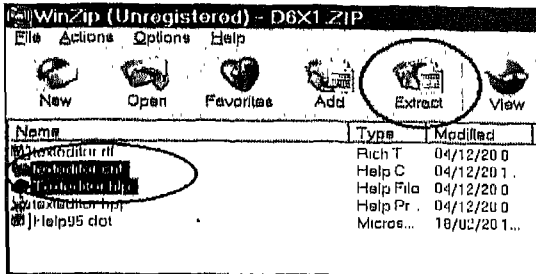


(٣) انقر بالماوس على المفتاح I Agree ليتم فتح نافذة البرنامج وبها نشاهد محتويات الملف D6X1.zip كما هو موضح بالشكل التالي :

شكل توضيحي :

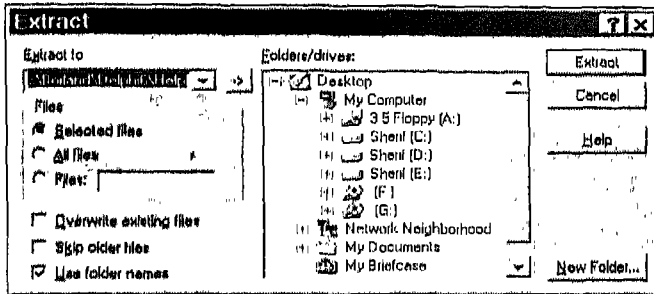


(٤) علم بالماوس على كل الملفات التي لها الامتداد .hlp والامتداد .cnt. ثم انقر بالماوس على المفتاح Extract كما هو موضح بالشكل التالي :



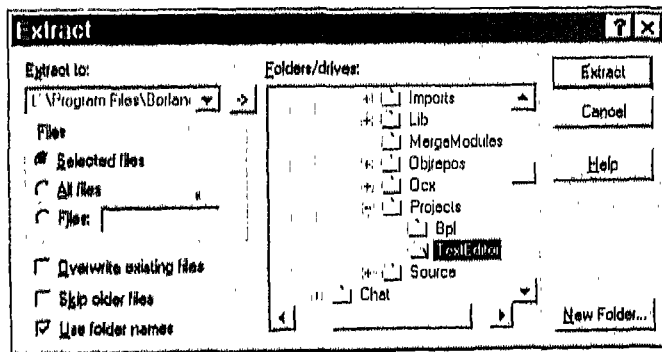
شكل توضيحي :

(٥) يظهر على الشاشة صندوق الحوار Extract الموضح في الشكل التالي :



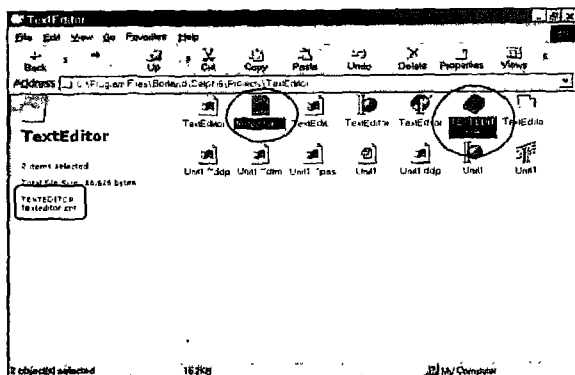
شكل توضيحي :

(٦) بقائمة العرض Folders/drives ابحث عن المجلد TextEditor الذي أنشأته في بداية هذا الفصل لتخزين ملف المشروع الذي تعدّه به. وهذا المجلد موجود بالمسار C:\Project Files\Borland\Delphi6\Projects\TextEditor كما هو موضح بالشكل التالي :



شكل توضيحي :

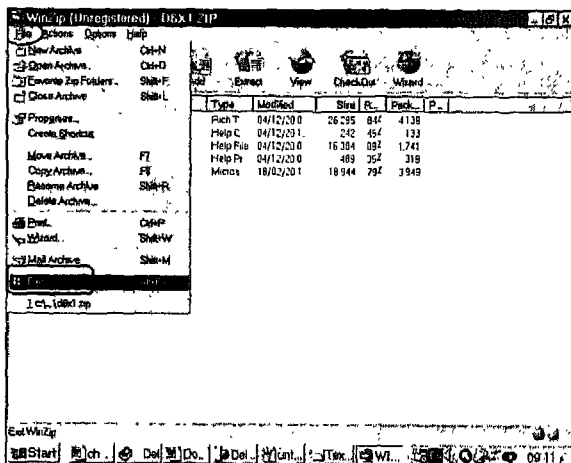
- (٧) انقر بالماوس على المفتاح Extract ليتم استخلاص الملفات المختارة ووضعها في المجلد المختار كما هو موضح بالشكل التالي :



شكل توضيحي :

- (٨) لقد اختفى صندوق الحوار Extract من على الشاشة وعدت مرة أخرى إلى نافذة البرنامج WinZip.

- (٩) قم بغلق البرنامج WinZip بفتح القائمة File واختيار الأمر Exit منها كما هو موضح بالشكل التالي :



شكل توضيحي :

تستطيع أن تستخدم أي ملف مساعدة له امتداد HLP أو CNT (مثل ملفات المساعدة HLP التي تأتي مع لغة Delphi والملفات CNT الملحقة



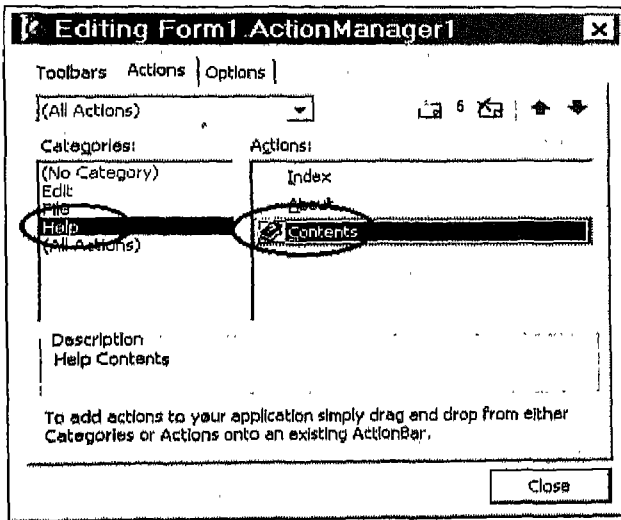


بها) بالمشروع الذي تعدده. وفي هذه الحالة ينبغي عليك تغيير اسماء هذه الملفات لتصب TextEditor.hlp و TextEditor.cnt وذلك لكي يتمكن التطبيق الذي تعدده من العثور على هذه الملفات.

إنشاء أداة معاملة الحدث الخاص بالأمر Help Contents

لإنشاء أداة معاملة الحدث الخاص بالأمر Contents بالقائمة Help اتبع الخطوات التالية :

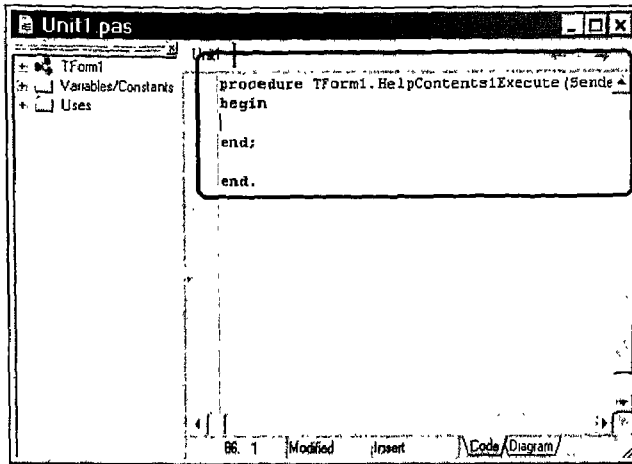
- (١) انقر بالماوس نقرا مزدوجا على أيقونة المكون ActionManager بالفورمة لفتح صندوق الحوار Action Manager (إذا لم يكن مفتوحا بالفعل).
- (٢) في صندوق الحوار Action Manager اختر Help من القائمة المنسدلة Categories ثم انقر بالماوس نقرا مزدوجا على الفعل Contents بقائمة العرض Actions كما هو موضح بالشكل التالي :



شكل توضيحي :

الآن يظهر على الشاشة محرر الكود البرمجي Code Editor وبه نشاهد مؤشر الكتابة داخل الهيكل البرمجي لأداة معاملة الحدث الخاص بالأمر Contents كما هو موضح بالشكل التالي :

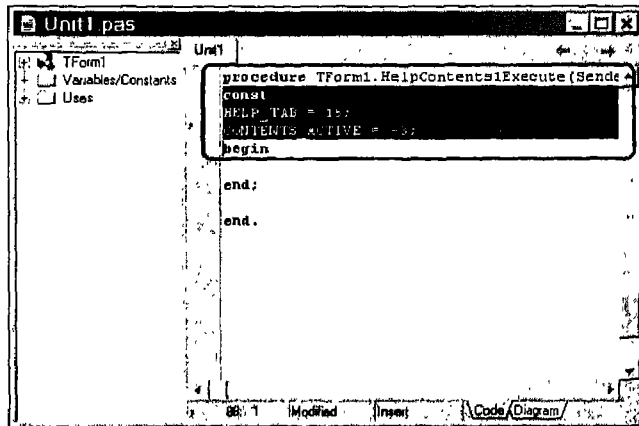
شكل توضيحي :



(٣) بالسطر الذي يسبق الكلمة begin مباشرة اكتب الكود التالي :

```
const
HELP_TAB = 15;
CONTENTS_ACTIVE = -3;
```

كما هو موضح بالشكل التالي :



شكل توضيحي :

ثم بالسطر الذي يلي الكلمة Begin مباشرة اكتب الكود التالي :

```
Application.HelpCommand(HELP_TAB, CONTENTS_ACTIVE);
```

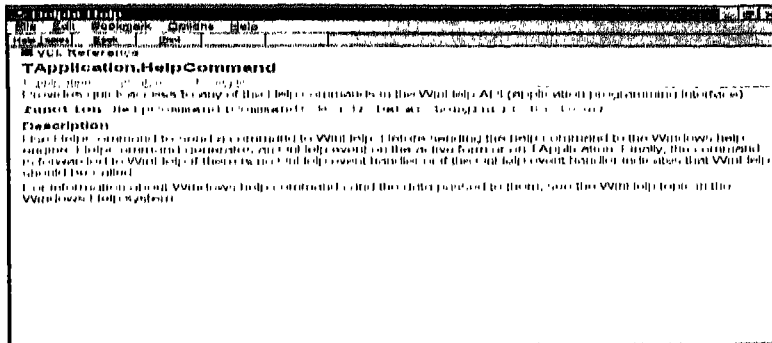
كما هو موضح بالشكل التالي :

شكل توضيحي :

```
Unit1
procedure TForm1.HelpContents1Execute(Sender: TObject);
const
  HELP_TAB = 15;
  CONTENTS_ACTIVE = -3;
begin
  Application.HelpCommand(HELP_TAB, CONTENTS_ACTIVE);
end;
end.
```

هذا الكود يعمل على تخصيص القيم الثابتة إلى المعاملات الخاصة بالأمر HelpCommand. ونود هنا القول بأن تخصيص ١٥ إلى المعامل HELP_TAB يؤدي إلى عرض صندوق الحوار Help كما أن تخصيص القيمة -٣ إلى المعامل CONTENTS_ACTIVE يؤدي إلى عرض التبويب Contents بصندوق الحوار Help.

إذا أردت على الحصول على المساعدة حول الحدث HelpCommand ضع مؤشر الكتابة بجوار HelpCommand في محرر الكود البرمجي Code Editor ثم اضغط على المفتاح F1 بلوحة المفاتيح ليظهر على الشاشة صندوق الحوار Help وبه الموضوع الذي ترغبه كما هو موضح بالشكل التالي :

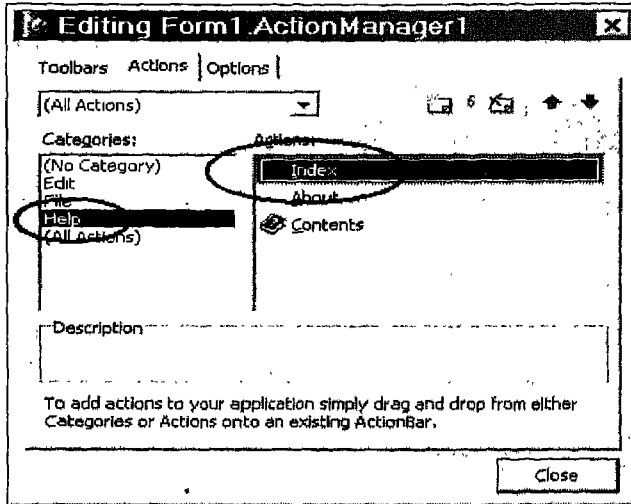


إنشاء أداة معاملة الحدث للأمر Help Index

لإنشاء أداة معاملة الحدث للأمر Help Index اتبع الخطوات التالية :

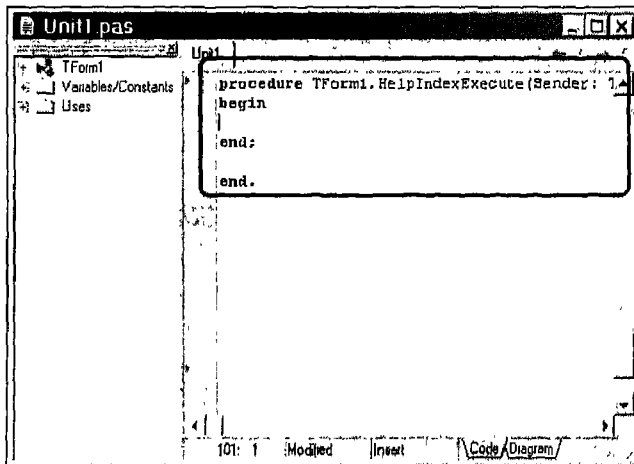
- (١) انقر بالماوس نقرا مزدوجا على أيقونة المكون ActionManager بالفورمة لفتح صندوق الحوار Action Manager (إذا لم يكن مفتوحا بالفعل).

(٢) في صندوق الحوار Action Manager اختر Help من القائمة المنسدلة Categories ثم انقر بالماوس نقرا مزدوجا على الفعل Index بقائمة العرض Actions كما هو موضح بالشكل التالي :



شكل توضيحي :

الآن يظهر على الشاشة محرر الكود البرمجي Code Editor وبه نشاهد مؤشر الكتابة داخل الهيكل البرمجي لأداة معاملة الحدث الخاص بالأمر Index كما هو موضح بالشكل التالي :



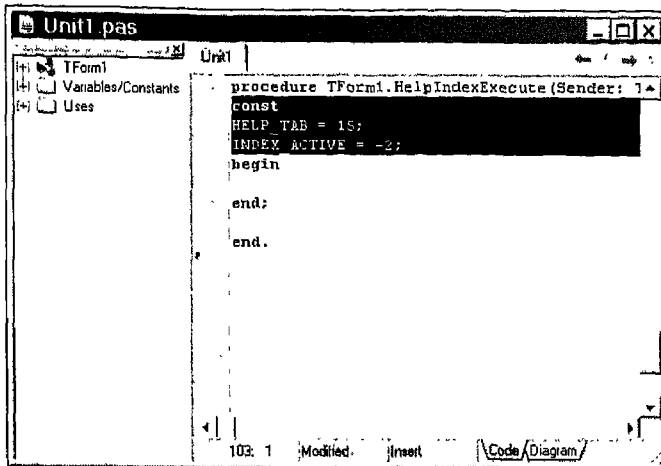
شكل توضيحي :

(٤) بالسطر الذي يسبق الكلمة begin مباشرة اكتب الكود التالي :



```
const
HELP_TAB = 15;
INDEX_ACTIVE = -2;
```

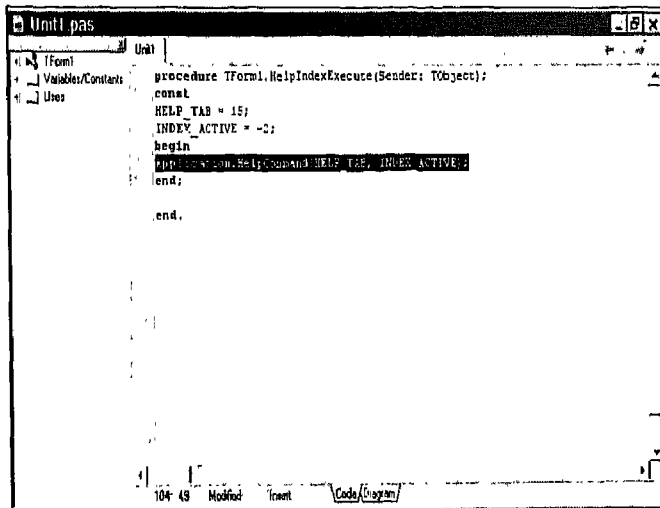
كما هو موضح بالشكل التالي :



شكل توضيحي :

ثم بالسطر الذي يلي الكلمة Begin مباشرة اكتب الكود التالي :
Application.HelpCommand(HELP_TAB, INDEX_ACTIVE);

كما هو موضح بالشكل التالي :



شكل توضيحي :

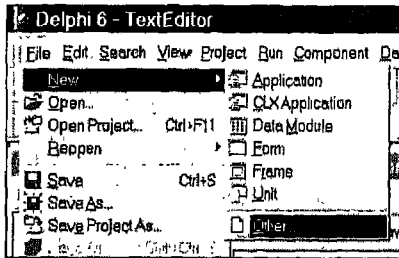
هذا الكود يعمل على تخصيص القيم الثابتة إلى المعاملات الخاصة بالأمر HelpCommand. ونود هنا القول بأن تخصيص ١٥ إلى المعامل HELP_TAB يؤدي إلى عرض صندوق الحوار Help كما أن تخصيص القيمة ٢- إلى المعامل CONTENTS_ACTIVE يؤدي إلى عرض التبويب Index بصندوق الحوار Help.

إنشاء صندوق الحوار About

العديد من التطبيقات تكون مشتملة على صندوق الحوار About الذي يعرض معلومات عن المنتج مثل اسمه ورقم الإصدار والعلامات التجارية logos كما قد يشتمل على معلومات عن الاستخدام القانوني للمنتج وحقوق الملكية. لقد إنتهيت حتى الآن من إضافة الأمر About للقسم Help بصندوق الحوار Action Manager.

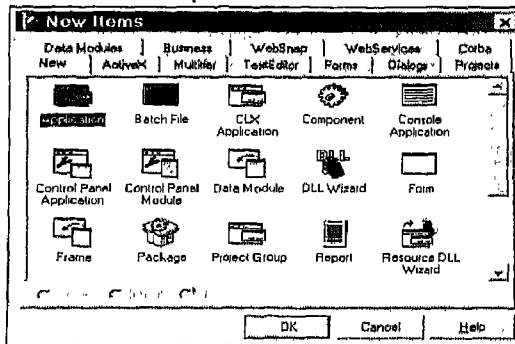
لإضافة صندوق الحوار About اتبع الخطوات التالية :

(١) افتح القائمة File ثم اختر New ثم اختر Other كما هو موضح بالشكل التالي :



شكل توضيحي :

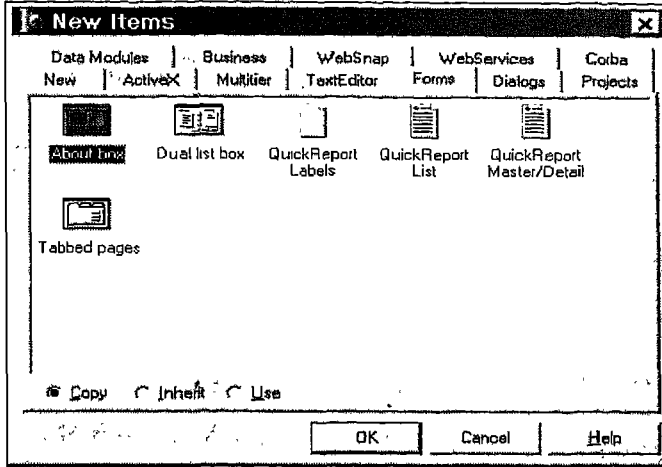
ليظهر على الشاشة صندوق الحوار New Items الموضح في الشكل التالي :



شكل توضيحي :

انقر بالماوس على التبويب Forms ليظهر على السطح كما هو موضح بالشكل

التالى :

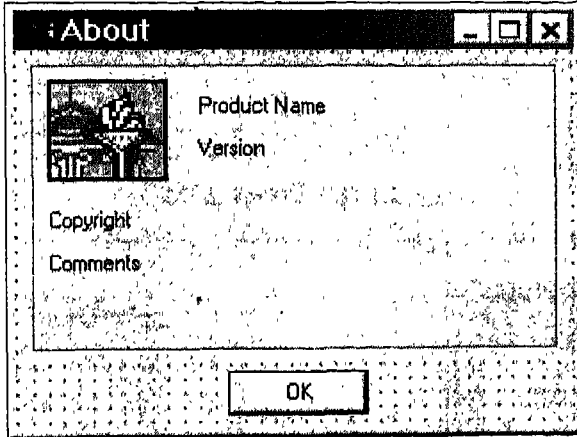


شكل توضيحي :

(٢) بالتبويب Forms انقر بالماوس نقرا مزدوجا على الأيقونة About Box ليتم

إنشاء فورمة جديدة تعمل على تبسيط عملية إنشاء صندوق الحوار About

والشكل التالى يوضح لنا هذه الفورمة الجديدة :



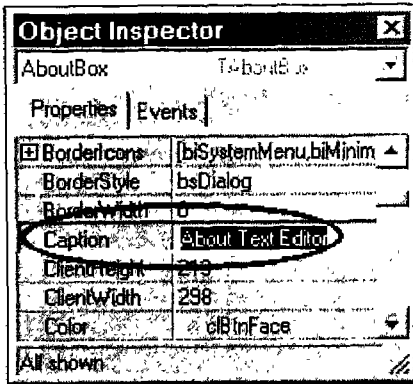
شكل توضيحي :

(٣) اختر الفورمة نفسها (بالنقر بالماوس على الجزء المنقط بها) ثم فى النافذة

Object Inspector اجعل قيمة الخاصية Caption عبارة عن About Text

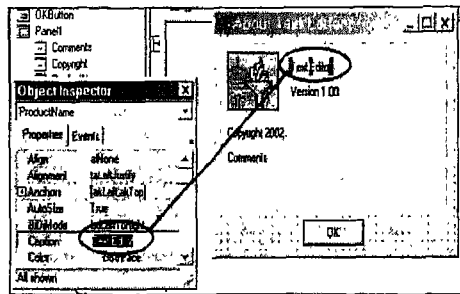
Editor كما هو موضح بالشكل التالى :

شكل توضيحي :



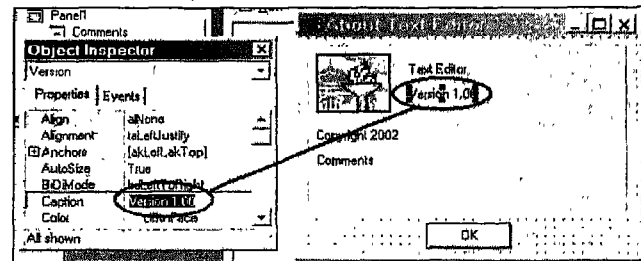
(٤) في هذه الخطوة ستقوم بتغيير النصوص (العناوين Labels) الموجودة بالفورمة About وذلك كالآتي :

انقر بالماوس على Product name بالفورمة About ثم في النافذة Object Inspector اجعل قيمة الخاصية Caption عبارة عن Text Editor كما هو موضح بالشكل التالي :



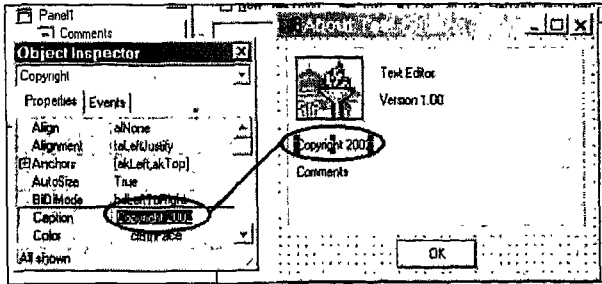
شكل توضيحي :

انقر بالماوس على Version بالفورمة About ثم في النافذة Object Inspector اجعل قيمة الخاصية Caption عبارة عن Version 1.00 كما هو موضح بالشكل التالي :



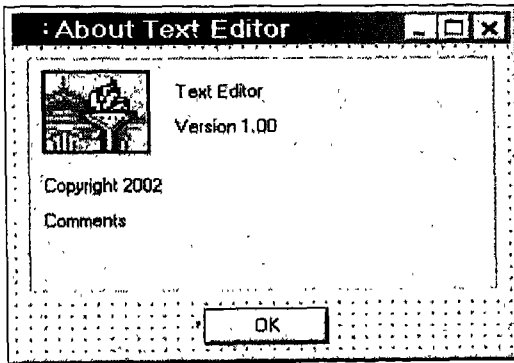
شكل توضيحي :

انقر بالماوس على Copyright بالفورمة About ثم في النافذة Object Inspector اجعل قيمة الخاصية Caption عبارة عن Copyright 2002 كما هو موضح بالشكل التالي :



شكل توضيحي :

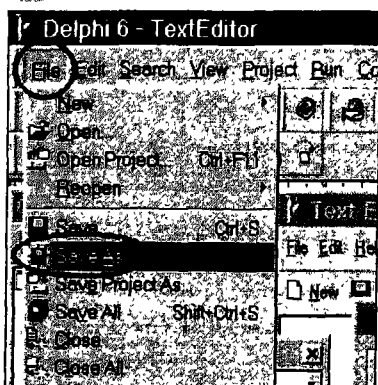
والآن تصبح الفورمة About كما هو موضح بالشكل التالي :



شكل توضيحي :

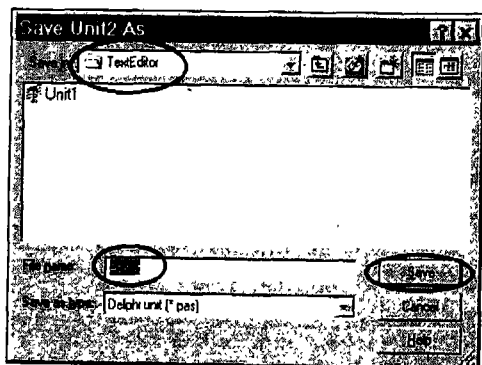
(٥) احفظ فورمة صندوق الحوار About بفتح القائمة File ثم اختيار الأمر Save As منها كما هو موضح بالشكل التالي :

شكل توضيحي :



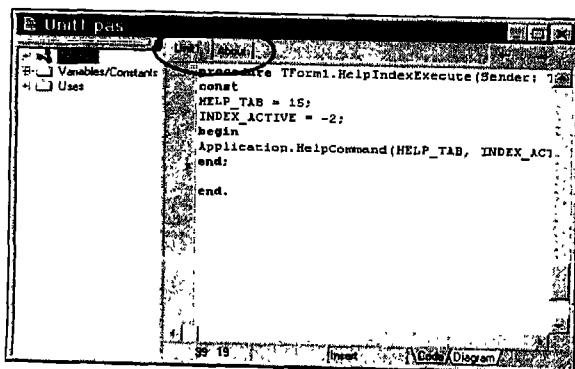
ليظهر على الشاشة صندوق الحوار Save Unit2 As ومن خلاله احفظ هذه الفورمة باسم About.pas كما هو موضح بالشكل التالي :

شكل توضيحي :



(٦) في محرر الكود البرمجي Code Editor بلغة Delphi ينبغي أن يكون به تبويب للوحدة Unit1 وتبويب آخر للوحدة About كما هو موضح بالشكل التالي :

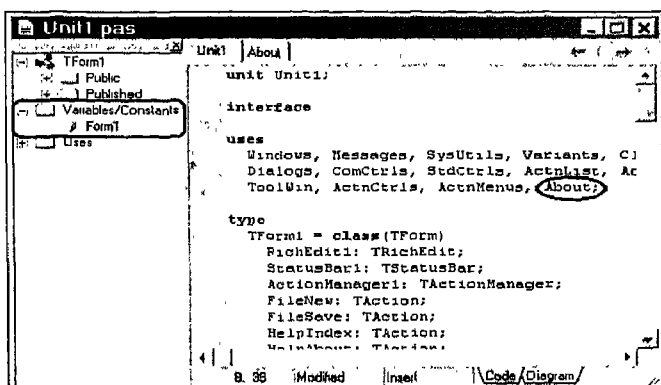
شكل توضيحي :



تلاحظ أن التبويب الخاص بالوحدة ActnRes غير موجود بمحرر الكود البرمجي وذلك لأنك لست في حاجة لهذه الوحدة.



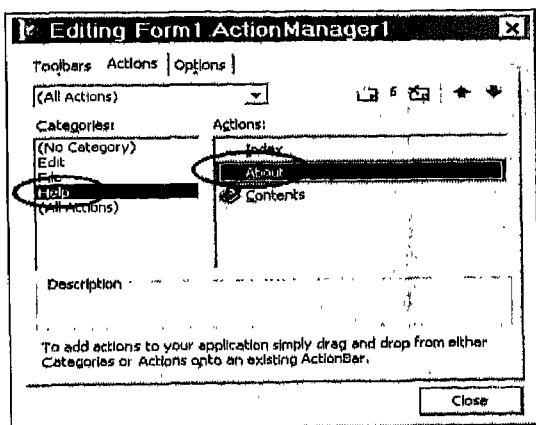
(٧) انقر بالماوس على التبويب Unit1 ثم اصف الوحدة الجديدة About بإضافة الكلمة About إلى أسماء الوحدات الموجودة ضمن الجملة uses كما هو موضح بالشكل التالي :



شكل توضيحي :

(٨) اضغط على المفتاح F12 بلوحة المفاتيح للعودة مرة أخرى إلى مود التصميم ثم انقر بالماوس نقرا مزدوجا على أيقونة المكون ActionManager لفتح صندوق الحوار Action Manager (إذا لم يكن مفتوح بالفعل).

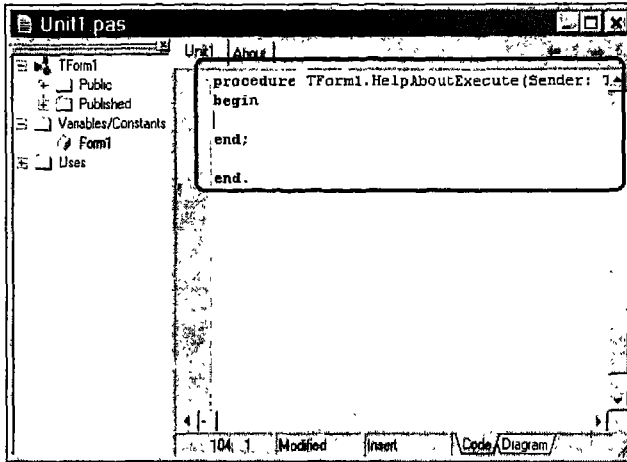
(٩) انقر بالماوس على القسم Help بقائمة العرض Categories ثم على الفعل Index بقائمة العرض Actions كما هو موضح بالشكل التالي :



شكل توضيحي :

انقر بالماوس نقرا مزدوجا على الفعل About ليتم إنشاء أداة معاملة حدث خاصة به وذلك في نافذة محرر الكود البرمجي Code Editor كما هو موضح بالشكل التالي :

شكل توضيحي :



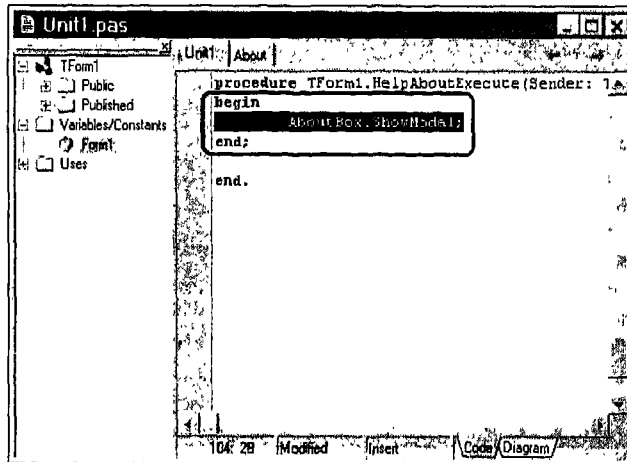
بالسطر الموجود به مؤشر الكتابة (بين الكلمة Begin والكلمة End) اكتب الكود

التالي :

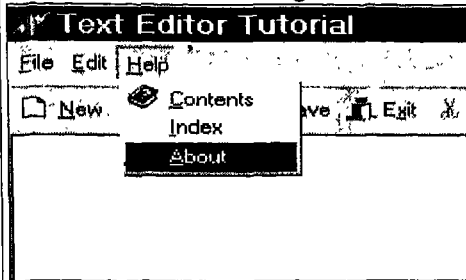
AboutBox.ShowModal;

كما هو موضح بالشكل التالي :

شكل توضيحي :



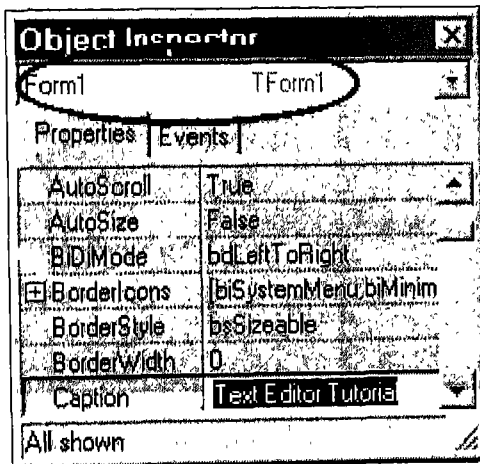
هذا الكود يفتح صندوق الحوار About عندما يقوم المستخدم بفتح القائمة Help ويختار منها الأمر About كما هو موضح بالشكل التالي :



استكمال التطبيق الذي تعدده

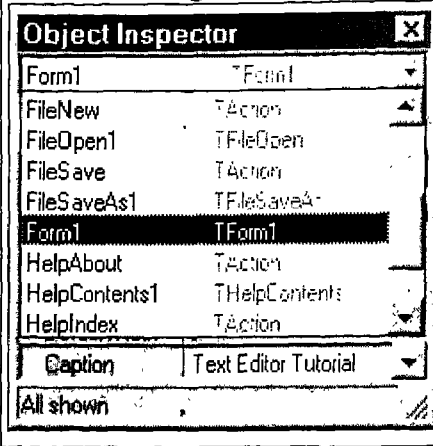
التطبيق الذي تتولى إعداده أصبح مكتملا إلى حد كبير. ولكن على كل حال ستظل مطالب بتحديد وتوصيف بعض العناصر في الفورمة الأساسية لهذا التطبيق. لاستكمال هذا التطبيق اتبع الخطوات التالية :

- (١) اضغط على المفتاح F12 بلوحة المفاتيح للتعامل مع الفورمة الأساسية بمود التصميم.
- (٢) تأكد من أن الفورمة نفسها هي المختارة وليس أحد العناصر الموجودة بها. ونود هنا القول بأن القائمة المنسدلة الموجودة في أعلى النافذة Object Inspector ينبغي أن تكون مشتملة على الاختيار Form1:TForm1 كما هو موضح بالشكل التالي :

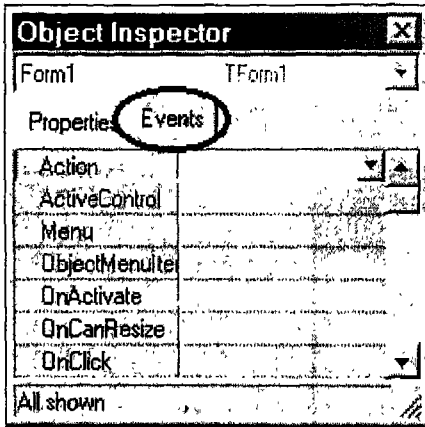


شكل توضيحي :

إذا لم تكن هذه القائمة المنسدلة مشتملة على هذا الاختيار في هذه الحالة افتحها واختر منها هذا الاختيار كما هو موضح بالشكل التالي :



(٣) انقر بالماوس على التبويب Events ليظهر على السطح بالنافذة Object Inspector كما هو موضح بالشكل التالي :



شكل توضيحي :

افتح القائمة المنسدلة الخاصة بالحدث OnCreate ومنها اختر FormCreate وذلك لإنشاء أداة معاملة حدث يصف ما الذي يحدث عندما تم إنشاء الفورمة (منذ أن قمت بفتح التطبيق).

(٤) يظهر على الشاشة محرر الكود البرمجي Code Editor وفيه اكتب الكود

التالي (بالسطر الموجود به مؤشر الكتابة بين Begin و End) :

```
Application.HelpFile := ExtractFilePath(Application.ExeName) +
'TextEditor.hlp';
FileNew.Execute
```

الهدف من هذا الكود الإعلان الابتدائي initializing عن التطبيق وذلك عن طريق

كل من الآتي :

● إلحاق ملف مساعدة.

● جعل قيمة الخاصية FileName عبارة عن untitled.txt.

● وضع اسم الملف بسطر الحالة.

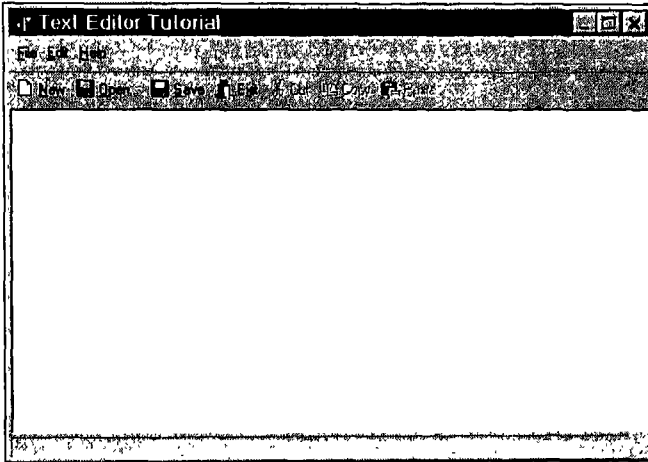
● تنظيف منطقة تحرير النص.

(٥) والآن افتح القائمة File ثم اختر منها الأمر Save All لتحفظ التغييرات التي

أجريتها بالمشروع.

(٦) اضغط على المفتاح F9 بلوحة المفاتيح لتشغيل البرنامج الذي أصبح مكتملا الآن

كما هو موضح بالشكل التالي :



شكل توضيحي :

الملحق

التحديث من الإصدارات السابقة
إلى Delphi 6



مقدمة عامة

عندما تستخدم Delphi 6 لتحميل مشروع تم إعداده من خلال Delphi 5 أو من خلال الإصدارات التي تسبقه من لغة Delphi في هذه الحالة يتم تحديث مكونات هذا المشروع تلقائياً. ونحن في هذا الفصل سنحاول سويًا دراسة التغييرات التي تؤثر بشكل قوى ومباشر على المشاريع التي تم إعدادها بالإصدارات السابقة من لغة Delphi عند التعامل معها من خلال Delphi 6. وهذه التغييرات يمكن حصرها في النقاط التالية :

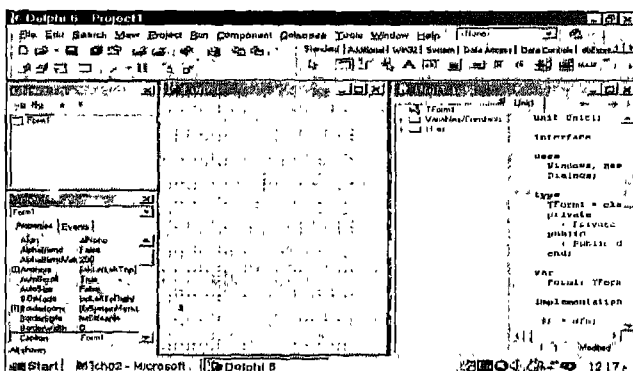
- التغييرات الخاصة ببيئة التطوير المتكاملة IDE.
- التحديثات التلقائية لأسماء الحزم البرمجية.
- التغييرات الخاصة بالتوافق مع الإصدارات السابقة.

يمكنك الرجوع إلى الفصل الأول "الجديد في Delphi 6" للحصول على المزيد من المعلومات حول المظاهر والإمكانيات الجديدة التي قد ترغب في التعامل معها بالتطبيقات التي تتولى إعدادها.



التغييرات الخاصة ببيئة التطوير المتكاملة IDE

عندما تفتح المشروع الذي تتولى إعداده وكان هذا الفتح لأول مرة من خلال Delphi 6 في هذه الحالة لن يبدو بنفس الشكل الذي كان يظهر به في الإصدارات السابقة من لغة Delphi. فهناك العديد من التغييرات التي تم إجراؤها على بيئة التطوير المتكاملة IDE بما فيها التغييرات التي أجريت على تخطيط النافذة وبالبيئة المكونات كما هو موضح بالشكل التالي :



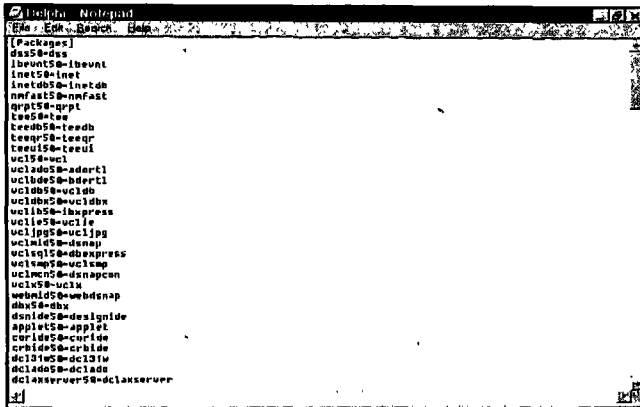
شكل توضيحي :

للمزيد من المعلومات حول هذه التغييرات يمكنك الرجوع لقسم المظاهر والإمكانيات الجديدة ببيئة التطوير المتكاملة IDE بالفصل الأول "الجديد في Delphi 6".



التحديثات التلقائية لأسماء الحزم البرمجية

كما هو الحال في التحديثات الأولية Prior للغة Delphi نجد أن أسماء العديد من الحزم البرمجية قد تغيرت. فعلى سبيل المثال قد تغير اسم الحزمة البرمجية vclide50 وأصبح vclide60. وفي هذا الصدد نقول إن الإصدارات السابقة للغة Delphi تقوم بأداء تحديثات تلقائية لأسم الحزمة البرمجية كذلك الحال بالنسبة هذا الإصدار الجديد من لغة Delphi والذي يقوم بنفس الشيء. ونود هنا القول بأن الآلية الخاصة بالتحديثات قد تغيرت تماماً فهي الآن تعتمد بشكل أساسي على قائمة تضم مجموعة الملفات التي تغيرت. هذا ويمكن يمكن العثور على هذه القائمة في الملف Delphi.upg الموجود في الفهرس Bin والموضح في الشكل التالي :



شكل توضيحي :

الطريقة السابقة تعتبر بديل لطريقة تغيير suffixes.



في الماضي عندما كانت لغة Delphi تشعر بأى خطأ يمكن أن يكون مرتبطاً بتغيير فى اسم أى من الحزم البرمجية فى هذه الحالة تقوم بالبحث عن أسماء الحزم البرمجية

وذلك من أجل العثور على Obselect suffixes ثم تقوم بشكل تلقائي باستبدال ما تعثر عليه بالـ Suffixes الصحيحة.

على سبيل المثال نجد أن الإصدار الخامس من لغة Delphi يقوم تلقائياً بتحديث الإشارة المرجعية لأحزمة البرمجية Package40 (في حالة وجود هذه الحزمة البرمجية) ومن ثم فإنه يقرأ الحزمة Package50 بدلا منها.

الإصدار السادس من لغة Delphi يؤدي المهمة السالفة الذكر وذلك عن طريق البحث بقائمة تضم أسماء الحزمة البرمجية Obselect ثم يقوم باستبدالهم كما هو مشار إليه بهذه القائمة.

هذه القائمة موجودة بالملف Delphi.upg الموجود بدوره في الفهرس Bin داخل المجلد الذي تم تركيب لغة Delphi 6 فيه.



يستطيع مستخدمو التطبيق مشاهدة وتغيير محتويات الملف Delphi.upg باستخدام أى برنامج من برامج تحرير النصوص البسيطة بما فيها محرر النصوص الخاص ببيئة التطوير المتكاملة IDE للغة Delphi كما هو موضح بالشكل التالى :



شكل توضيحي :

ومثل هذه الإمكانية قد تكون مفيدة جدا وخصوصا لمصممي المكونات والكائنات وذلك لكونهم يرغبوا فى تعديل المكونات التى يقوموا بتصميمها بحيث تصبح متوافقة مع Delphi 6 والاحتفاظ بالمكونات فى حزمة برمجية مختلفة.



التغييرات الخاصة بالتوافق مع الإصدارات السابقة

فيما يلي نقط التوافق العامة التي قد تؤثر بشكل مباشر في التطبيقات التي تعدها بلغة Delphi :

- مصدر توفير البيانات والأحداث المرتبطة بمجموعة بيانات العميل والتي تتأثر بالتغير في البناء الهيكلي للقطاع VCL.
- التغيير المطلوب إجراؤه بالكود البرمجي من أجل صندوق حوار الولوج لقاعدة البيانات Database Login (الافتراضي).
- عدم التوافق بين ملفات البيانات الثنائية الخاصة بالفورم.
- التغيير في الثوابت القابلة للكتابة Writeable.
- Unary negation لنوع البيانات Cardinal.
- إعادة تسمية DsgnIntf والتغييرات المرتبطة به.
- التغييرات التي أجريت على محرر المكونات.
- التغييرات التي أجريت على المكون TDesignWindow.
- التغييرات التي أجريت على الحزمة البرمجية VCL.
- انتقال وحدة الوسيط OpenGL إلى rtl.dcp.
- الأنواع التي انتقلت من الوحدة HTTPApp.pas إلى الوحدة HTTPProd.pas.
- إزالة وحدة البحث وانتقال وتغيير SearchBuf.

هناك بعض الموضوعات الخاصة في الفصل الأول "الجديد في Delphi 6" قد تؤثر بشكل أساسي ومباشر في توافق التطبيقات التي تعدها مع الإصدار ٦ للغة Delphi ومن بينها ما يلي :

- المظاهر الجديدة لمترجم اللغة.
- المظاهر الجديدة للقطاع VCL (على الأخص الجزء المسمى "التغييرات والوحدات المضافة").

مصدر توفير البيانات وأحداث مجموعة بيانات العميل المتأثرة بالتغيير في البناء الميكلي للقطاع VCL

التقديم للمكون TCustomClientDataSet يتطلب إجراء تغييرات على أدوات معالجة الأحداث بالأكواد البرمجية التي تم كتابتها من خلال الإصدار الخامس من لغة Delphi والإصدارات السابقة له.

هناك ستة أحداث تنتمي إلى خمسة أنواع تأثرت بالتغيير التي أجرى على الوحدة DBCLIENT.PAS وأنواع هذه الأنواع والأحداث عبارة عن الآتي :

النوع	الحدث والتغيير
TResolveErrorEvent	يؤثر على الحدث OnUpdateError الخاص بمصدر توفير البيانات.
TBeforeUpdateRecordEvent	يؤثر على الحدث BeforeUpdateRecord الخاص بمصدر توفير البيانات.
TAfterUpdateRecordEvent	يؤثر على الحدث AfterUpdateRecord الخاص بمصدر توفير البيانات.
TProviderDataEvent	يؤثر على كل من الحدث OnGetData والحدث OnUpdateData الخاص بمصدر توفير البيانات.
TReconcileErrorEvent	يؤثر على الحدث OnReconcile الخاص بقاعدة بيانات العميل.

عند استخدام أدوات المعاملة الخاصة بالأحداث المذكورة في الجدول السابق ينبغي عليك استبدال TClientDataSet بـ TCustomClientDataSet.





التغيير المطلوب إجراؤه بالكود البرمجي لصندوق الحوار الافتراضي Database Login

فى السابق كان تحديد قيمة الخاصية LoginPrompt لأى مكون من مكونات الاتصال (مثل المكون TDatabase أو المكون TADOConnection والمكون TDCOMConnection) يجعل صندوق الحوار الافتراضى Database Login يظهر على الشاشة. ولكن الآن لم يعد هذا التحديد يؤدى إلى ظهور صندوق الحوار هذا وذلك إذا لم تقوم بإضافة DBLogDlg إلى الجملة Uses بالكود البرمجي الذى تكتبه. وفى هذا الصدد نقول إن التطبيقات التى تعتمد على صندوق الحوار الافتراضى Database Login يجب تعديل الكود البرمجي الخاص بها لإضافة DBLogDlg إلى الجملة Uses وإلا فلن تقوم هذه التطبيقات بعرض صندوق الحوار هذا والذى يطلب اسم المستخدم وكلمة المرور.

عدم التوافق بين ملفات البيانات الثنائية الخاصة بالفورم

فى الماضى كانت الإصدارات الأقدم من لغة Delphi لديها القدرة على قراءة الملفات الثنائية للفورم (أو الملفات DFM) التى يتم إنشاؤها بالإصدارات الأحدث من لغة Delphi. ولكن هذه الإمكانية لم تعد متاحة فى Delphi 6 فهناك بعض ملفات الفورم الثنائية التى يمكن قرائتها بطريقة غير صحيحة وذلك بسبب الطريقة التى يتبعها Delphi 6 لأداء عملية Streaming الداخلية للسلاسل الحرفية.

هذا وفى الماضى كانت عملية Streaming تتم مفترضة مجموعة حروفيات محلية معينة. ولكن الآن نجد أن عملية Streaming تفترض أن مجموعة الحروفيات عبارة عن UTF-8. ومن ثم لو كانت هناك حروفيات مكودة بأكواد أكبر من ١٢٧ (مثل رمز حقوق الاستخدام ©) فى الملف الثنائى للفورمة المعدة بلغة Delphi 6 فإم مثل هذا الملف لن تتم قراءته بواسطة الإصدارات الأقدم للغة Delphi.

لو أنك تنوى أن تستخدم ملف فورمة معدة بـ Delphi 6 (أو ملفات فورم معدة بإصدارات سابقة للغة Delphi تم إستيرادها وتعديلها من خلال Delphi 6) فى إصدار أقدم من لغة Delphi فى هذه الحالة ينبغى حفظ هذا الملف بتنسيق نصى بدلا من التنسيق الثنائى.



**التغيير في الثوابت القابلة للكتابة Writeable**

الحالة الافتراضية لمحول مترجم اللغة \$WRITEABLECONST (aka \$J) أصبحت الآن OFF مما سيؤدي إلى منع المشاريع المعدة بلغة Delphi من أن يكون لديها ثوابت من النوع Writeable. وهذه النوعية من الثوابت تشير إلى استخدام لثابت محدد النوع على أساس كونه متغير قابل للتعديل في مرحلة التشغيل run-time. وفيما يلي مثال لذلك :

```
const .
    foo: Integer = 12;
begin
    foo := 14;
end.
```

في الإصدارات السابقة للغة Delphi كان يعد ذلك Construct مقبولا ولم تكن الثوابت في حقيقة الأمر ثوابت. ولكن مع كون المحول \$WRITEABLECONST لا يعمل OFF فإن هذا الكود لن يجعل أى مترجم يعطى رسالة خطأ عند تعامله مع جملة التخصيص للمتغير foo في البلوك Begin...End. ولإصلاح هذا عليك أن تقوم بتغيير الإعلان عن الثابت ليصبح إعلان عن متغير var.

من الممكن أن يكون لديك كود برمجى يستخدم النوع Const للإعلان المبدئى عن متغير محلى بعمر افتراضى عام global lifetime كما هو موضح بالكود البرمجى التالى :

```
procedure MyProc;
const
    somedata: Integer = 12;
begin
    Inc(somedata, 3);
end;
```

ستحتاج لأن تنقل المتغير الذى ينتمى للنوع const إلى خارج الإجراء والإعلان عنه على أساس أنه متغير عام. وبعد إجراء هذا التغيير سيصبح الكود البرمجى السابق كالاتى :

```
var
    somedata: Integer = 12;
```



```
procedure MyProc;
```

```
begin
```

```
    Inc(somedata, 3);
```

```
end ;
```

عندما يكون الكود البرمجي معتمدا بشدة على النوع Const Quirk (مثل أداة تكوين أدوات التحكم ActiveX) في هذه الحالة يمكن إدراج الموجهة {\$WRITEABLECONST ON} بملف المصدر المشتمل على الكود البرمجي على أساس أن ذلك حل سريع. ومثل هذا العمل محظور في كل من RTL و VCL و CLX والكود البرمجي المصدرى لقواعد البيانات DB كما إنه يعتبر عمل غير مشجع في المصادر IDE ولكنه مقبول بالنسبة للوحدات الثانوية مثل الـ Wrappers الخاصة بأدوات التحكم ActiveX.

بصفة عامة ينبغي عليك ملاحظة أن عبارة "ثابت قابل للكتابة Writeable constant" ما هي إلا إجتماع لكلمتين على طرفى نقيض تماما. فالإصدارات السابقة للغة Delphi كانت لهم بشكل افتراضى للإبقاء على مستوى التوافق ثابت بدون أى خلل مع النسخ القيمة من مترجم اللغة والتي كانت تتعامل مع أنظمة التشغيل الـ 16Bit لم تعد مهمة لأغلب المبرمجين بلغة Delphi.

ننصحك دوما بأن تتبع أسلوب برمجي جيد وحاول أن تتفادى الثوابت التي من النوع Writeable.



Unary negation لنوع البيانات Cardinal

فى الماضى كانت لغة Delphi تتعامل مع Unary negation لأرقام تنتمى لنوع البيانات Cardinal وذلك باستخدام معاملات التشغيل الخاصة بأنظمة التشغيل الـ ٣٢ بت مما يؤدي إلى الحصول على بعض النتائج الفردية.

هنا مثال لكود برمجي يستخدم Unary negation :

```
var
```

```
    c: Cardinal;
```

```
    i: Int64;
```

```
begin
```



```
c := 4294967294;
i := -c;
WriteLn(i);
end ;
```

في الإصدارات السابقة للغة Delphi نجد أن قيمة *i* التي يتم عرضها عبارة عن ٢ وهذا من الواضح سلوك غير صحيح تماما بالنسبة لهذه الحالة. ولكن في Delphi 6 يتم التعامل مع Unary negation بعد السماح للنوع Cardinal بأى يكون نوع له إشارة ٦٤ بت ومن ثم القيمة النهائية لـ *i* والتي يتم عرضها تكون -4294967294.

إعادة تسمية DsgnIntf والتغييرات المرتبطة به

الإشارات المرجعية إلى DsgnIntf في المشروع الذى تتولى إعداده يجب أن تتغيرت إلى الأسم الجديد DesignIntf بالإصدار السادس للغة Delphi. كذلك قد تحتاج إلى إضافة DesignEditors و VCLEditors و RTLConsts بالجملة Uses بالكود البرمجى الذى تكتبه. كذلك ستحتاج لأن تضيف DesignIde إلى قائمة متطلبات الحزمة البرمجية التى تتولى إعدادها. أما الإشارات المرجعية References إلى dsnd50 فينبغى أيضا تغييرها إلى DesignIde لو أن هذا التغيير لم يتم تلقائيا بواسطة لغة Delphi نفسها.

أى حزمة برمجية خاصة بمرحلة التشغيل runtime packages تستخدم IDesigner تحتاج لأن تستخدم IDesignerHook لكى تتفادى متطلبات designide فى مرحلة التشغيل. هذا وفى الكود البرمجى لمرحلة التشغيل يجب أن يكون IDesignerHook وافيا للغرض. أما الكود البرمجى لمرحلة التصميم فيمكن أن يستخدم IDesigner ولكن ينبغى أن يستخدم شيء ما يشبه الآتى :

```
var
  RealDesigner: IDesigner;
...
  SomeDesignerHook.QueryInterface(IDesigner, RealDesigner);
...
```

وذلك للحصول على واجهة استخدام (وسيط) IDesigner حقيقية من أى حالة من حالات IDesignerHook يتطلب فقط قطاعات وفورم لكى يكون متاحا للاستخدام. هذا ويحتاج



الـ IDesigner إلى DesignIntf والذي يتضمن العديد من الحزم البرمجية الأخرى بعض منها قد يكون غير قابل لإعادة النشر والتوزيع مرة أخرى.

التغييرات التي أجريت على محرر المكونات

القطاع TComponentEditor لديه سلسلة نسب مختلفة في Delphi 6. ففي Delphi 5 كان هذا القطاع ينحدر من القطاع TInterfaceObject ولكنه الآن ينحدر من القطاع الجديد TBaseComponentEditor. كذلك فإن القطاع TComponentEditorClass أصبح الآن قطاع ينحدر من TBaseComponentEditor وذلك بدلا من أن ينحدر من القطاع TComponentEditor. وهذه التغييرات في البناء الهيكلي قد تتطلب منك أن تقوم بتعديل المشاريع التي صممتها من خلال الإصدارات السابقة للغة Delphi.

التغييرات التي أجريت على القطاع TDesignWindow

لقد تم إجراء عدد من التغييرات المرتبطة بالقطاع TDesignWindow. فلقد تم نقله إلى الوحدة DesignWindows كما أن الأسلوب FormClosed الخاص به قد تم استبداله بالأسلوب الجديد DesignerClosed. وفي الماضي كان أى شخص يستطيع الوصول إلى الفورمة من خلال الأسلوب FormClosed وذلك باستخدام المعامل AForm الخاص بهذا الأسلوب. ولكن مع الأسلوب الجديد DesignerClosed أصبح من الضروري الآن استخدام الخاصية Root الخاصة بـ Designer للوصول إلى الفورمة.

من خلال الأسلوب FormClosed كان أى شخص يستطيع إنشاء قوائم اختيار عن طريق استدعاء الخاصية Create للمكون TDesignerSelectionList أو الخاصية Create للمكون TComponentList. أما الآن فلكى يتم إنشاء قوائم اختيار من خلال الأسلوب الجديد DesignerClosed فمن الضروري ان يتم استخدام وسيط IDesignerSelections. وأنت تستطيع إنشاء واحد من هذه الأوساط باستخدام الالة CreateSelectionList.

المعاملات الخاصة بالأسلوب SelectionClosed أصبحت الآن تختلف عما كانت عليه في Delphi 5.



**التغييرات التي أجريت على الحزم البرمجية VCL**

محتويات بعض الحزم البرمجية المرتبطة بـ VCL قد تم إعادة توزيعها من خلال حزم برمجية أخرى. ولو أنك قمت بإعداد إشارات مرجعية إلى الوحدة vcl50.dcp بالمشروع الذى تتولى إعداده فى هذه الحالة ستحتاج لتغيير هذه الإشارات المرجعية إلى وحدات أخرى مثل الوحدة vcl.dcp والوحدة rtl.dcp.

انتقال وحدة الوسيط OpenGL إلى rtl.dcp

وحدة الوسيط OpenGL التى أعدتها شركة Borland (الموجودة بالملف Opengl.dcu) كانت فى الماضى وحدة مستقلة بذاتها فى Delphi 5 وكانت موجودة بالمجلد Lib. ولكن الآن فقد تم ضمها داخل الملف rtl.dcp وذلك فى Delphi 6. ومثل هذا التغيير قد يتسبب فى حدوث بعض المشاكل بالنسبة للمشاريع التى تم إعدادها من خلال Delphi 5 والتى يتم فتحها وتشغيلها من داخل Delphi 6.

لتوضيح ما سبق سنذكر المثال التالى :

فى مشروع معد بـ Delphi 5 كان من الممكن عمل تخطى بطريقة معينة للوحدة OpenGL وذلك عن طريق وضع وحدة تكون بنفس الأسم فى مكان ما بالمسار الخاص بالمشروع. ولكن عند استخدام نفس الطريقة فى Delphi 6 فإن ذلك يتسبب فى حدوث الخطأ name conflict فى أى مكون يستخدم rtl.dcp ومن ثم يكون من المطلوب الآن تغيير الاسم.

الأنواع التى انتقلت من الوحدة HTTPApp.pas إلى الوحدة HTTPProd.pas

لقد تم نقل العديد من الأنواع الموجودة فى الوحدة HTTPApp لتصبح الآن فى الوحدة HTTPProd. وهذه الأنواع عبارة عن THTMLBgColor و THTMLAlign و THTMLVAlign. ولو كانت المشاريع التى تعدها تستخدم أى من هذه الوحدات فى هذه الحالة ينبغى عليك تغيير الجمل uses الموجودة بالكود البرمجية لهذه المشاريع بحيث تشير إلى الوحدة HTTPProd بدلا من الوحدة HTTPApp.



إزالة وحدة البحث وإنقال وتغيير الروتين SearchBuf

لم تعد الوحدة Search موجودة في Delphi 6. أما بالنسبة للروتين SearchBuf والذي يعمل على تحديد موضع سلسلة حرفية فرعية داخل مخزن للنصوص Text Buffer قد تم نقله إلى الوحدة StrUtils كما أن المعاملات الخاصة بهذا الروتين قد تغيرت هي الأخرى. والمعامل النهائي أصبح الآن الكائن TStringSearchOptions. ولو أن المشروع الذى تتولى إعداده لن تتم ترجمته من خلال Delphi 6 بسبب أن المترجم لا يستطيع العثور على الوحدة Search فى هذه الحالة قم بتغيير الجملة uses بحيث تضم StrUtils بدلا من Search. كما إنك ستحتاج أيضا فى مراجعة الاستدعاءات التى أعددتها إلى SearchBuf وذلك لكى تتأكد من أنك المعاملات التى تستخدمها تتطابق مع الصياغة الجديدة.

١	الفصل الأول : الجديد فى Delphi ٦
٣	ما هو الجديد فى Delphi ٦؟
٤	المظاهر والتحسينات الجديدة ببيئة التطوير المتكاملة IDE
٤	الModules الخاصة بالبيانات (كافة الإصدارات)
٥	أداة المشاهدة الشجرية TreeView للكائن (كافة الإصدارات)
٨	محرف الكود البرمجى
٨	أدوات تصميم السطح (الإصدار الفنى Professional والإصدار المتكامل Enterprise)
٨	صفحة الديجرام (الإصدار الفنى Professional والإصدار المتكامل Enterprise)
٩	صفحات الWebSnap (الإصدار المتكامل Enterprise)
١٠	تبويبات السحب والإسقاط (كافة الإصدارات)
١٠	أداة فحص الكائن Object Inspector (كافة الإصدارات)
١٠	قائمة عرض اللحظية Instance list box
١١	صندوق حوار الخصائص
١٢	مرجعيات المكون الممتدة Inline
١٣	أدوات التعامل مع الكود البرمجى Code insight tools (كافة الإصدارات)
١٣	تكملة الكود البرمجى Code completion
١٤	معاملات الكود البرمجى code parameters
١٤	الmodule الجديد لتخطيط المفاتيح Key mapping (كافة الإصدارات)
١٥	الحزم Packages فى مدير المشروع (كافة الإصدارات)
١٥	قائمة الملف File Menu
١٦	صندوق حوار العناصر الجديدة New Items
١٦	التبويب WebSnap :
١٧	التبويب Web Services :
١٧	التبويب CORBA :
١٨	شريط أدوات الإنترنت (الإصدار المتكامل Enterprise)
١٩	التغييرات التى أجريت على باليطة المكونات
٢٠	صندوق حوار خيارات بيئة العمل Environment Options
٢٠	تبويب المصمم Designer :

٢١	تبويب متغيرات بيئة العمل Environment Variables :
٢١	تبويب أداة فحص الكائن Object Inspector :
٢٢	تبويب الإنترنت Internet :
٢٢	صندوق حوار الفهارس Directories
٢٢	عرض قائمة المحتوى Context menu
٢٣	كتابة الحزم البرمجية في أثناء مرحلة التصميم Design-time packages
٢٣	مظاهر وإمكانيات الإنترنت الجديدة (الإصدار الفنى Professional والإصدار المتكامل Enterprise)
٢٣	الدعم الخاص بخدمات الويب (الإصدار المتكامل Enterprise فقط)
٢٣	الدعم الخاص بتطبيقات خادم الويب (الإصدار الفنى Professional والإصدار المتكامل Enterprise)
٢٤	خادم الويب Apache (الإصدار الفنى Professional والإصدار المتكامل Enterprise)
٢٤	أداة معالجة الثغرات لتطبيقات الويب Web Application Debugger (الإصدار الفنى والإصدار المتكامل)
٢٤	المظاهر والإمكانيات WebBroker المحسنة (الإصدار الفنى Professional والإصدار المتكامل Enterprise)
٢٤	مظاهر WebSnap الجديدة (الإصدار المتكامل Enterprise فقط)
٢٥	Modules الويب المتعددة
٢٥	المعالجات الجديدة لـ Module الويب
٢٧	إعداد Scripts للعمل بخوادم الويب
٢٧	المكونات الجديدة
٢٧	مكونات أداة تحقيق الأفعال Dispatcher
٢٧	مكونات المنظم Adapter Components
٢٨	مكونات أداة إنتاج صفحة الويب Page Producer Components
٢٨	مكونات Session
٢٨	مكونات قائمة المستخدم User List
٢٩	أدوات تصميم السطح بـ WebSnap
٢٩	الدعم الخاص بلغة XML (الإصدار المتكامل Enterprise فقط)
٢٩	معالج ربط بيانات الـ XML
٢٩	برمجة المستندات المعدة بلغة XML
٣٠	برمجة الكائن DOM للعمل عبر أنظمة التشغيل المختلفة
٣٠	استخدام XML في تطبيقات قواعد البيانات
٣٠	المظاهر الجديدة لمترجم اللغة Compiler

فهرس الكتاب

٣٠Variants المتغيرات
٣١ Enumerated أنواع البيانات الرقمية أو العددية
٣١ Consts Unit الثوابت على وحدة الثوابت
٣١ directives الجديدة لدى مترجم اللغة
٣٢ Conditional directives الموجهات الشرطية
٣٢ \$If الموجهة \$If
٣٢ \$ALIGN حقل الضبط \$ALIGN
٣٢ built-in assembler الجديدة أداة التجميع الأساسية
٣٣ التحديد الافتراضى الجديد للثوابت القابلة للتحديد
٣٣ تقديم الدعم لأنواع الملفات المختلفة
٣٣ Overload الموجهات التى أجريت على وظيفة التحميل الزائد
	المظاهر الجديدة لأدوات التحكم ActiveX والنموذج COM (الإصدار الفنى
٣٤ Professional والإصدار المتكامل Enterprise)
٣٤ COM التسجيل والتركيب لصفات تهيئة النموذج
٣٤ Event Object Wizard المعالج
٣٥ Interfaces الموجودة بالفعل تنفيذ الـ
٣٥ Transactional الكائنات الحركية (كبدل للمعالج MTS)
٣٦ MTS/COM+ المزود للـ الكائنات الحركية الدعم
٣٦ Enterprise Professional والإصدار المتكامل Enterprise) المظاهر الجديدة لقواعد البيانات
٣٦ dbExpress آلية الوصول للبيانات
٣٨ أنواع الحقول الجديدة
٣٩ Client Dataset الهياكل البنائية لمجموعة بيانات العميل
٤١ multi-tiered الدعم الخاص بالتطبيقات المتعددة الطبقات
٤٢ الموجهات التى أجريت على بالليقة المكونات
٤٤ CORBA (الإصدار المتكامل Enterprise فقط) المظاهر والإمكانات الجديدة
٤٦ Actions الجديدة (الإصدار الفنى Professional والإصدار المتكامل Enterprise) مظاهر وإمكانات الأفعال
٤٦ Action Bands حزم الأفعال
٤٨ الأفعال القياسية الجديدة
٤٩ التحسينات التى أضيفت لقطاعات الأفعال

٤٩	المظاهر والإمكانات الجديدة للوحدات VCL الجديدة (كافة الإصدارات)
٥٠	المكونات الجديدة
٥٠	المظاهر والإمكانات التي تم تطويرها
٥١	المكونات الفرعية Subcomponents
٥١	خصائص الـ Interface القابلة للنشر
٥١	الإضافات والتغييرات التي أجريت لوحدات البرمجة
٦٠	الوحدات RTL الجديدة والمظاهر الجديدة الخاصة بها (كافة الإصدارات)
٦٢	تدعيم أنواع المتغيرات الخاصة (كافة الإصدارات)
	إمكانية إعداد تطبيقات تعمل عبر مختلف نظم التشغيل (الإصدار الفني Professional والإصدار المتكامل Enterprise)
٦٣	الاختلافات في القطاع المكتبي CLX
٦٤	إنشاء تطبيق يعمل بمختلف أنظمة التشغيل cross-platform application
٦٥	أدوات الترجمة والمطورة (الإصدار المتكامل Enterprise)
٦٦	التغييرات التي أجريت على طريقة نشر وتوزيع التطبيقات (كافة الإصدارات)
٦٨	زيادة كافة نظام المساعدة (كافة الإصدارات)
٧١	الفصل الثاني : أدوات العمل ببيئة التطوير المتكاملة للغة Delphi ٦
٧٣	القائمة File
٧٣	الأمر New القائمة File
٧٤	الأمر Application القائمة New بالقائمة File
٧٤	الأمر CLX Application القائمة New بالقائمة File
٧٤	الأمر Data Module القائمة New بالقائمة File
٧٥	الأمر Form القائمة New بالقائمة File
٧٥	الأمر Frame القائمة New بالقائمة File
٧٥	الأمر Unit القائمة New بالقائمة File
٧٦	الأمر Other القائمة New بالقائمة File
٧٦	صندوق الحوار New Items
٧٦	الصفحة New بصندوق الحوار New Items
٨٤	الصفحة ActiveX بصندوق الحوار New Items
٩٢	الصفحة Multitier بصندوق الحوار New Items

فهرس المحتاآ

٩٥	New Items	الآوار	Project	الصفآ
٩٥	New Items	الآوار	Forms	الصفآ
٩٩	New Items	الآوار	Dialogs	الصفآ
١٠٢	New Items	الآوار	Projects	الصفآ
١٠٦	New Items	الآوار	Data Modules	الصفآ
١٠٧	New Items	الآوار	Business	الصفآ
١١٠	New Items	الآوار	WebSnap	الصفآ
١١٢	New Items	الآوار	Web Services	الصفآ
١١٤	New Items	الآوار	CORBA	الصفآ
١١٥	File	بالقائمة	Open	الأمر
١١٦	File	بالقائمة	Open Project	الأمر
١١٦	File	بالقائمة	Reopen	الأمر
١١٧	File	بالقائمة	Save	الأمر
١١٧	File	بالقائمة	Save As	الأمر
١١٧	File	بالقائمة	Save Project As	الأمر
١١٨	File	بالقائمة	Save All	الأمر
١١٨	File	بالقائمة	Close	الأمر
١١٨	File	بالقائمة	Close All	الأمر
١١٨	File	بالقائمة	Use Unit	الأمر
١١٩	File	بالقائمة	Print	الأمر
١١٩	File	بالقائمة	Exit	الأمر
١١٩	Edit	بالقائمة		القائمة
١٢١	Edit	بالقائمة	Undo/Delete	الأمر
١٢١	Code Editor	البرمآى	Undo	استآءام الأمر فى محرر الكود البرمآى
١٢٣	Edit	بالقائمة	Redo	الأمر
١٢٤	Edit	بالقائمة المختصرة	Cut	الأمر (أو بالقائمة المختصرة بآافذ محرر الكود البرمآى)
١٢٤	Edit	بالقائمة المختصرة	Copy	الأمر (أو بالقائمة المختصرة بآافذ محرر الكود البرمآى)
١٢٥	Edit	بالقائمة المختصرة	Paste	الأمر (أو بالقائمة المختصرة بآافذ محرر الكود البرمآى)
١٢٥	Edit	بالقائمة	Delete	الأمر

١٢٦	Edit القائمة Select All الأمر
١٢٦	Edit القائمة Align to Grid الأمر (أو من القائمة المختصرة بالفورمة)
١٢٨	Edit القائمة Bring to Front الأمر (أو بالقائمة المختصرة بالفورمة)
١٢٨	Edit القائمة Send to Back الأمر (أو بالقائمة المختصرة بالفورمة)
١٢٨	Edit القائمة Align الأمر
١٢٩	Alignment صندوق حوار الضبط
١٣٠	Edit القائمة Size الأمر
١٣١	Size صندوق الحوار
١٣٢	Edit القائمة Scale الأمر (أو بالقائمة المختصرة الخاصة بالفورمة) صندوق الحوار Scale
١٣٢	معامل التكبير والتصغير (نسبة مئوية)
١٣٣	Edit القائمة Tab Order الأمر (أو بالقائمة المختصرة بالفورمة)
١٣٤	Edit Tab Order صندوق الحوار
١٣٤	Edit Tab Order صندوق الحوار Controls القائمة العرض
١٣٥	Edit القائمة Creation Order الأمر (بالقائمة المختصرة بالفورمة)
١٣٦	Creation Order صندوق الحوار
١٣٧	Edit القائمة Flip Children الأمر (بالقائمة المختصرة بالفورمة)
١٣٨	Edit القائمة Flip Children الأمر All بقائمة الأمر
١٣٨	Edit القائمة Flip Children الأمر Selected بقائمة الأمر
١٣٨	Edit القائمة Lock Controls الأمر
١٣٩	Edit القائمة Add to Interface الأمر
١٤٠	القائمة Search
١٤١	Find القائمة الأمر
١٤١	Find Text صندوق الحوار
١٤٢	Find Text صندوق الحوار الخاص بالخيارات
١٤٢	Text to find الحقل
١٤٤	Search Find في الملفات القائمة الأمر
١٤٤	Find Text صندوق الحوار
١٤٥	Text to Find القائمة المنسدلة
١٤٦	Options القائمة الموجودة في القسم

١٤٧ Search Directory Options	الاختيارات الموجودة بالقسم
١٤٩ Search Replace	الأمر Replace بالقائمة
١٤٩ Replace Text	صندوق الحوار
١٥٠ Text to Find	حقل القائمة المنسدلة
١٥٠ Replace With	حقل القائمة المنسدلة
١٥١ Options	الخيارات الموجودة فى القسم
١٥٣ Replace All	المفتاح
١٥٣ Search Again	الأمر Search بالقائمة
١٥٣ Incremental Search	الأمر Incremental Search بالقائمة
١٥٥ Go to Line Number	الأمر Go to Line Number بالقائمة
١٥٥ Go To Line Number	صندوق الحوار
١٥٦ Enter New Line Number	حقل القائمة المنسدلة
١٥٦ Search Find Error	الأمر Find Error بالقائمة
١٥٧ Find Error	صندوق الحوار
١٥٧ Error Address	حقل القائمة المنسدلة
١٥٧ Search Browse Symbol	الأمر Browse Symbol بالقائمة
١٥٨ Browse Symbol	صندوق الحوار
١٥٩ View	القائمة
١٦٠ View Project Manager	الأمر Project Manager بالقائمة
١٦٢ View Translation Manager	الأمر Translation Manager بالقائمة
١٦٢ View Object Inspector	الأمر Object Inspector بالقائمة
١٦٣ View Object TreeView	الأمر Object TreeView بالقائمة
١٦٥ View To-Do List	الأمر To-Do List بالقائمة
١٦٥ View Alignment Palette	الأمر Alignment Palette بالقائمة
١٦٦ View Browser	الأمر Browser بالقائمة
١٦٩ View Code Explorer	الأمر Code Explorer بالقائمة
١٧١ View Component List	الأمر Component List بالقائمة
١٧٢ Components	نافذة المكونات
١٧٤ Window List (أو القائمة Window View)	الأمر Window List (بالقائمة View أو القائمة Window)

١٧٤	Window List صندوق الحوار
١٧٥	View الأمر Debug Windows بالقائمة
١٧٦	View الأمر Breakpoints بالقائمة الفرعية Debug Windows من القائمة
١٧٧	View الأمر Call Stack بالقائمة الفرعية Debug Windows من القائمة
١٧٨	View الأمر Watches بالقائمة الفرعية Debug Windows من القائمة
١٧٨	View الأمر Local Variables بالقائمة الفرعية Debug Windows من القائمة
١٧٩	Local Variables القائمة المختصرة الخاصة بنافذة المراقبة
١٨١	View الأمر Threads بالقائمة الفرعية Debug Windows من القائمة
١٨٢	View الأمر Modules بالقائمة الفرعية Debug Windows من القائمة
١٨٥	إجراء عملية debugging لأكثر من عملية في نفس الوقت
١٨٦	View الأمر Event Log بالقائمة الفرعية Debug Windows من القائمة
١٨٨	View الأمر CPU بالقائمة الفرعية Debug Windows من القائمة
١٨٩	View الأمر FPU بالقائمة الفرعية Debug Windows من القائمة
١٩١	View الأمر Desktops بالقائمة
١٩٢	View الأمر Save Desktop بالقائمة الفرعية للأمر Desktops بالقائمة
١٩٣	View الأمر Delete بالقائمة الفرعية للأمر Desktops بالقائمة
١٩٥	View الأمر Set Debug Desktop بالقائمة الفرعية للأمر Desktops بالقائمة
١٩٦	View الأمر Toggle Form/Unit بالقائمة
١٩٦	View الأمر Units بالقائمة
١٩٧	View Unit صندوق الحوار
١٩٧	View الأمر Forms بالقائمة
١٩٨	View form صندوق الحوار
١٩٨	View الأمر Type Library بالقائمة
١٩٩	View الأمر New Edit Window بالقائمة
٢٠٠	View الأمر Toolbars بالقائمة
٢٠٢	Standard Toolbar شريط الأدوات القياسي
٢٠٤	View Toolbar شريط الأدوات
٢٠٥	Debug Toolbar شريط الأدوات
٢٠٧	Custom Toolbar شريط الأدوات

فهرس الكتاب

٢٠٨ Desktops Toolbar شريط الأدوات
٢٠٨ Internet Toolbar شريط الأدوات
٢١١ External Editor استخدام المحرر الخارجي
٢١١ External Editor خطوات إعداد وتهيئة المحرر الخارجي
٢١٣ External Editor خطوات فتح المحرر الخارجي
٢١٣ Project Menu القائمة
٢١٥ Project Add to Project الأمر بالقائمة
٢١٥ Add To Project صندوق الحوار
٢١٨ Project Remove from Project الأمر بالقائمة
٢١٩ Project Import Type Library الأمر بالقائمة
٢١٩ Project Add To Repository الأمر بالقائمة
٢٢٠ Project View Source الأمر بالقائمة
٢٢٠ Project Languages الأمر بالقائمة
٢٢١ Project Languages الفرعية للأمر بالقائمة
٢٢٣ Project Languages الفرعية للأمر بالقائمة
٢٢٤ Project Languages الفرعية للأمر بالقائمة
٢٢٥ Project Languages الفرعية للأمر بالقائمة
٢٢٦ Project Add New Project الأمر بالقائمة
٢٢٧ Project Add Existing Project الأمر بالقائمة
٢٢٧ Project Compile Project الأمر بالقائمة
٢٢٧ Project Build Project الأمر بالقائمة
٢٢٨ Project Syntax check Project الأمر بالقائمة
٢٢٨ Project Information for Project الأمر بالقائمة
٢٢٩ Project Compile All Projects الأمر بالقائمة
٢٢٩ Project Build All Projects الأمر بالقائمة
٢٢٩ Project Web deployment Options الأمر بالقائمة
٢٣١ Project Web Deploy الأمر بالقائمة
٢٣١ Project Options الأمر بالقائمة
٢٣٥ Run القائمة

٢٤٥	خطوات إضافة نقاط توقف من خلال مشاهد اكتشاف ومعالجة الأخطاء debugging views
٢٤٦	القائمة Component
٢٤٩	القائمة Database
٢٥٤	القائمة Tools
٢٦٩	الخيارات الإضافية بالقائمة Tools
٢٧٣	الفصل الثالث : استخدام مكتبة المكونات Components لدى Delphi
٢٧٥	بيئة التطوير المتكاملة IDE
٢٧٧	تصميم التطبيقات
٢٧٩	استخدام مكتبة المكونات
٢٨١	الخصائص والأساليب والأحداث
٢٨١	الخصائص Properties
٢٨٣	الأساليب Methods
٢٨٣	الأحداث Events
٢٨٣	أنواع الأحداث
٢٨٤	أحداث المستخدم User Events
٢٨٤	أحداث النظام System Events
٢٨٤	الـ Object Pascal ومكتبات القطاعات
٢٨٧	استخدام نموذج الكائن Object Module
٢٨٧	ما هو الكائن Object
٢٨٨	اكتشاف الكائنات المتاحة لدى لغة Delphi
٢٩٤	تغيير اسم مكون
٢٩٧	توارث البيانات والكود البرمجي من كائن
٣٠١	مدى ومجال الاستخدام Scope وأدوات التأهيل Qualifiers
٣٠٣	الإعلانات الخاصة والمحمية والعامة والمنشورة
٣٠٤	استخدام متغيرات الكائن Object Variables
٣٠٥	إنشاء وتدمير الكائنات
٣٠٧	المكونات والملكية Ownership
٣٠٨	الكائنات والمكونات وأدوات التحكم
٣١٠	فرع قطاع التأسيس TObject

فهرس الكتاب

٣١٢	Tpersistent	فرع قطاع التأسيس
٣١٣	Tcomponent	فرع قطاع التأسيس
٣١٦	Tcontrol	فرع قطاع التأسيس
٣١٨	TwinControl	فرع قطاع التأسيس
٣٢٠	Tcontrol	الخصائص المشتركة لأدوات التحكم المنحدرة من قطاع التأسيس
٣٢٠	Action Properties	خصائص الأفعال
٣٢١	Alignment	خصائص تحديد الموضع والحجم والضبط
٣٢٢	Display Properties	خصائص العرض
٣٢٢	Parent Properties	خصائص الأب
٣٢٣	Navigation Property	خاصية التجول
٣٢٣	Drag-and-Drop	خصائص السحب والإسقاط
٣٢٤	Drag-and-Dock	خصائص السحب والإرساء
٣٢٧		الفصل الرابع : تمرين عملي (إنشاء تطبيق بسيط لتحرير النصوص)
٣٢٩		مقدمة عامة
٣٢٩		البدء في تطبيق جديد
٣٣٣		تحديد قيم الخصائص
٣٣٦		إضافة المكونات إلى الفورمة
٣٤٢		إضافة الدعم الخاص بالقوائم وشرائط الأدوات
٣٤٦	Action Manager	إضافة الأفعال إلى مدير الأفعال
٣٥٢		إضافة الأفعال القياسية لقائمة الأفعال
٣٥٧	Image List	إضافة الصور لقائمة الصور
٣٦٤	Menu	إضافة قائمة للمشروع
٣٦٨	toolbar	إضافة شريط أدوات للمشروع
٣٧٥		إخلاء منطقة النص من محتوياتها
٣٧٦	Event handlers	كتابة أدوات معاملة الأحداث
٣٨١	Open	إنشاء أداة معاملة الحدث المرتبط بالأمر
٣٨٩	Save	إنشاء أداة معاملة الحدث للأمر
٣٩١	Save As	إنشاء أداة معاملة الحدث المرتبط للأمر
٣٩٧		إنشاء ملف مساعدة

٤٠٢	إنشاء أداة معاملة الحدث الخاص بالأمر Help Contents
٤٠٤	إنشاء أداة معاملة الحدث للأمر Help Index
٤٠٧	إنشاء صندوق الحوار About
٤١٤	استكمال التطبيق الذي تعدّه
٤١٧	الملحق رقم (١) : التحديث من الإصدارات السابقة إلى Delphi ٦
٤١٩	مقدمة عامة
٤١٩	التغييرات الخاصة ببيئة التطوير المتكاملة IDE
٤٢٠	التحديثات التلقائية لأسماء الحزم البرمجية
٤٢١	التغييرات الخاصة بالتوافق مع الإصدارات السابقة
٤٢٣	مصدر توفير البيانات وأحداث مجموعة بيانات العميل المتأثرة بالتغير في البناء الهيكلي للقطاع VCL
٤٢٤	التغيير المطلوب إجراؤه بالكود البرمجي لصندوق الحوار الافتراضي Database Login
٤٢٤	عدم التوافق بين ملفات البيانات الثنائية الخاصة بالفورم
٤٢٥	التغيير في الثوابت القابلة للكتابة Writable
٤٢٦	Unary negation لنوع البيانات Cardinal
٤٢٧	إعادة تسمية DsgnIntf والتغييرات المرتبطة به
٤٢٨	التغييرات التي أجريت على محرر المكونات
٤٢٨	التغييرات التي أجريت على القطاع TdesignWindow
٤٢٩	التغييرات التي أجريت على الحزم البرمجية VCL
٤٢٩	انتقال وحدة الوسيط OpenGL إلى rtl.dcp
٤٢٩	الأنواع التي انتقلت من الوحدة HTTPApp.pas إلى الوحدة HTTPProd.pas
٤٣٠	إزالة وحدة البحث وانتقال وتغيير الروتين SearchBuf

لقد

أعدنا هذا الكتاب من أجل أن

نتوجه به إلى المبرمجين عامة المبتدئين منهم خاصة الذين

يخطون خطواتهم الأولى في مجال البرمجة من خلال الإصدار السادس للغة Delphi. هذا ونود هنا القول بأن هناك فجوة كبيرة بين الإصدارات السابقة للغة Delphi وهذا الإصدار الجديد الذي جعل لغة Delphi بحق لديها القدرة على منافسة أقوى لغات البرمجة التي تنتمي للطائفة للبرنامج (Object Oriented Programming) مثل لغة ++C و #C و Java و VC++

من خلال Delphi 6 يمكننا الآن إعداد تطبيقات قواعد بيانات تعمل في شبكات الحاسب الآلي الضخمة بالإضافة إلى إمكانية إعداد التطبيقات التي تعمل بخوادم شبكة الويب فضلا عن إمكانية إعداد التطبيقات المكتبية desktop Application وتطبيقات الموزعة Distributed Applications وتطبيقات الوسائط المتعددة. لقد رأينا أن تكون طريقة الشرح سهلة وبسيطة ومدعمة بالكثير من الأشكال التوضيحية والأمثلة العملية لكي تتناسب مع مستوى المبرمجين جميعا وخصوصا المبتدئين

والله الموفق

الناشر



دار الكتب العلمية للنشر والتوزيع

٥٠ شارع الشيخ ربحان - عابدين - القاهرة

I.S.B.N 977-287-277-3

٧٩٥٤٢٢٩ ☎